

**CA-IDMS®**

---

Conversion Notebook  
15.0



Computer Associates™

This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

THIS DOCUMENTATION MAY NOT BE COPIED, TRANSFERRED, REPRODUCED, DISCLOSED, OR DUPLICATED, IN WHOLE OR IN PART, WITHOUT THE PRIOR WRITTEN CONSENT OF CA. THIS DOCUMENTATION IS PROPRIETARY INFORMATION OF CA AND PROTECTED BY THE COPYRIGHT LAWS OF THE UNITED STATES AND INTERNATIONAL TREATIES.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

THE USE OF ANY PRODUCT REFERENCED IN THIS DOCUMENTATION AND THIS DOCUMENTATION IS GOVERNED BY THE END USER'S APPLICABLE LICENSE AGREEMENT.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227.7013(c)(1)(ii) or applicable successor provisions.

**First Edition, December, 2000**

© 2000 Computer Associates International, Inc.  
One Computer Associates Plaza, Islandia, NY 11749  
All rights reserved.

All trademarks, trade names, service marks, or logos referenced herein belong to their respective companies.

# Contents

---

<b>How to use this manual</b>	xi
<b>Chapter 1. Introduction</b>	1-1
1.1 About this chapter	1-3
1.2 CA-IDMS conversion activities	1-4
1.3 Release 12.0 test environment	1-11
What the test environment includes	1-11
Review dictionary and DMCL descriptions	1-11
1.4 Before you begin	1-12
<b>Chapter 2. Physical Database Definition</b>	2-1
2.1 About this chapter	2-3
Convert DMCL before migrating dictionary	2-3
Conversion tools and environment	2-3
Related CA documentation	2-4
2.1.1 An overview of the process	2-4
What's next	2-5
2.2 Convert 10.2 DMCL definitions	2-6
12.0 physical database components	2-6
The DMCL Syntax Generator	2-6
Sequence of statements	2-6
Before you run the DMCL Syntax Generator	2-6
2.2.1 Execute the DMCL Syntax Generator	2-7
Purpose	2-7
Authorization	2-7
Syntax	2-8
Parameters	2-8
Usage	2-9
2.2.2 IDMSDMCC compiler batch execution JCL	2-9
2.2.2.1 OS/390 execution JCL	2-9
2.2.2.2 VSE/ESA execution JCL	2-10
2.2.2.3 VM/ESA commands	2-11
2.2.2.4 BS2000/OSD execution JCL	2-11
2.2.3 What the DMCL Syntax Generator produces	2-12
CREATE SEGMENT statement	2-12
CREATE FILE statement	2-13
CREATE AREA statement	2-13
2.2.3.1 DMCL statements	2-14
CREATE DMCL statement	2-14
CREATE BUFFER statement	2-15
CREATE JOURNAL BUFFER statement	2-15
CREATE JOURNAL statements	2-16
CREATE DISK JOURNAL statement	2-16
CREATE ARCHIVE JOURNAL statement	2-16
CREATE TAPE JOURNAL	2-16
2.2.3.2 Segment definitions within the DMCL	2-16
ALTER DMCL statement	2-16

Transaction summary . . . . .	2-17
2.2.4 Correct errors from IDMSDMCC . . . . .	2-18
2.3 Create 12.0 DMCL module . . . . .	2-19
Review the output from the IDMSDMCC compiler . . . . .	2-19
What to review . . . . .	2-19
2.4 Create a 12.0 database name table . . . . .	2-23
Why you need a database name table . . . . .	2-23
What's next . . . . .	2-23
2.4.1 Specifying the default dictionary . . . . .	2-23
What is the default dictionary . . . . .	2-23
Default dictionary identified as a DBNAME . . . . .	2-23
2.4.2 Grouping segments . . . . .	2-24
Segment name as DBNAME . . . . .	2-24
Multiple segments associated with a DBNAME . . . . .	2-24
Examples . . . . .	2-24
2.4.3 Specifying a DBNAME at run time . . . . .	2-25
Options for specifying DBNAME at run time . . . . .	2-25
No DBNAME specified during bind processing . . . . .	2-25
2.4.4 Guidelines for defining database name tables . . . . .	2-25
2.4.5 Basic DBTABLE definition . . . . .	2-25
2.4.6 Secondary dictionaries . . . . .	2-27
Sharing dictionary areas . . . . .	2-27
Different page groups . . . . .	2-27
2.4.7 Conflicting area names . . . . .	2-27
Used 10.2 DMCL ALIAS parameter . . . . .	2-27
Used DBNAMES in 10.2 . . . . .	2-27
Subschema naming standards . . . . .	2-27
2.4.8 Non-dictionary DBNAMEs . . . . .	2-28
Review 10.2 subschema mapping rules . . . . .	2-28
DBNAME specified externally in 10.2 . . . . .	2-28
Example . . . . .	2-28
Redirecting run unit to different DBNAME . . . . .	2-29
2.4.9 Mixed page groups . . . . .	2-29
<b>Chapter 3. Security Migration . . . . .</b>	<b>3-1</b>
3.1 About this chapter . . . . .	3-3
The CA-IDMS central security facility . . . . .	3-3
Security definition migration . . . . .	3-3
3.2 About security definition migration . . . . .	3-4
What is the security definition migration utility . . . . .	3-4
What the security migration utility does . . . . .	3-4
What the security migration utility produces . . . . .	3-4
How to execute the security definition migration utility . . . . .	3-5
Conversion tools and environment . . . . .	3-5
Related CA documentation . . . . .	3-5
3.3 Convert security definitions . . . . .	3-6
3.4 Create execution JCL for the RHDCSMIG program . . . . .	3-7
Runs in local mode or under the central version . . . . .	3-7
Input to RHDCSMIG . . . . .	3-7
3.4.1 OS/390 JCL . . . . .	3-7
Local mode . . . . .	3-8

Central version	3-9
3.4.2 VSE/ESA JCL	3-9
Central version	3-10
3.4.3 VM/ESA commands	3-10
Central version	3-11
3.4.4 BS2000/OSD JCL	3-11
Local mode	3-11
Central version	3-12
3.5 Output from RHDCSMIG	3-13
CREATE RESOURCE SYSTEM statement	3-13
CREATE USER statement	3-13
CREATE SYSTEM PROFILE	3-13
GRANT SIGNON ON SYSTEM ... TO USER	3-14
CREATE RESOURCE CATEGORY	3-14
GRANT EXECUTE ON CATEGORY	3-15
CREATE RESOURCE ACTIVITY	3-17
GRANT EXECUTE ON ACTIVITY	3-18
3.6 Review and modify output from RHDCSMIG	3-20
3.7 Populate 12.0 system with security definitions	3-22
Submit statements to the command facility	3-22
Connect to the system dictionary	3-22
Review listing from the command facility	3-22
Security Display Facility	3-22
<b>Chapter 4. Dictionary Migration and Setup</b>	4-1
4.1 About this chapter	4-3
4.2 About the 12.0 dictionary environment	4-4
Migrate only the DDLDML area	4-4
Unchanged dictionary areas	4-4
New dictionary areas	4-4
4.3 About dictionary migration	4-5
The dictionary migration utility	4-5
What the dictionary migration utility does	4-5
When to migrate dictionaries	4-5
Conversion tools and environment	4-5
Related CA documentation	4-6
4.4 Planning to migrate dictionaries	4-7
4.5 Migrate dictionaries	4-8
4.5.1 The dictionary migration process	4-8
What's next	4-8
4.5.2 Running RHDCMIG1	4-8
4.5.2.1 OS/390 execution JCL	4-8
4.5.2.2 VSE/ESA execution JCL	4-9
4.5.2.3 VM/ESA commands	4-10
4.5.2.4 BS2000/OSD execution JCL	4-11
4.5.3 Verify the results of RHDCMIG1	4-12
Possible errors from RHDCMIG1	4-12
Abend code	4-12
4.5.4 Running RHDCMIG2	4-12
4.5.4.1 OS/390 JCL	4-13

What is an external user session . . . . .	5-11
What is an external request unit . . . . .	5-11
Specifying MAXIMUM ERUS parameter . . . . .	5-12
Specifying execution characteristics . . . . .	5-12
The RHDCNP3S task . . . . .	5-12
5.3.5 Regenerate your modified system definition . . . . .	5-12
5.4 Prepare your DC/UCF environment for startup . . . . .	5-13
5.5 Review and modify startup components . . . . .	5-14
Reassemble the #DCPARM macro . . . . .	5-14
System startup components . . . . .	5-14
<b>Chapter 6. Utilities</b> . . . . .	6-1
6.1 About this chapter . . . . .	6-3
6.2 Utility statements . . . . .	6-4
Utility statements that replace programs . . . . .	6-4
PUNCH statement . . . . .	6-5
Execution JCL . . . . .	6-5
6.3 Utility programs . . . . .	6-6
Status of utility programs . . . . .	6-6
Changes in existing programs . . . . .	6-6
Removal of IDMSRNWK . . . . .	6-7
Local mode execution JCL . . . . .	6-7
6.4 Security for utilities . . . . .	6-8
<b>Chapter 7. User Exits and Database Procedures</b> . . . . .	7-1
7.1 About this chapter . . . . .	7-3
7.2 User exits . . . . .	7-4
7.2.1 New user exits . . . . .	7-4
User exit 3 is replaced . . . . .	7-4
Control block changes . . . . .	7-4
7.3 Database procedures . . . . .	7-5
<b>Chapter 8. Extent Areas</b> . . . . .	8-1
8.1 About this chapter . . . . .	8-3
DDLCDQUE area . . . . .	8-3
ASF extent areas . . . . .	8-3
8.2 Migrate ASF extent areas . . . . .	8-4
Procedures . . . . .	8-4
8.2.1 Prepare your environment to migrate extent areas . . . . .	8-4
Release 10.2 environment . . . . .	8-4
Release 12.0 environment . . . . .	8-5
8.2.2 Run IDMSRSSM program . . . . .	8-5
Purpose . . . . .	8-5
Syntax . . . . .	8-5
Parameter . . . . .	8-6
8.2.2.1 Execution JCL . . . . .	8-6
OS/390 JCL . . . . .	8-6
VSE/ESA JCL . . . . .	8-7
VM/ESA commands . . . . .	8-8
BS2000/OSD JCL . . . . .	8-9

8.2.3	Output from IDMSRSSM . . . . .	8-9
	What IDMSRSSM produces . . . . .	8-9
	Sample IDMSRSSM output listing . . . . .	8-10
	Review the IDMSRSSM output listing . . . . .	8-11
8.2.4	Generate the IDMSRSSD subschema . . . . .	8-13
8.2.5	Unload and reload ASF data areas . . . . .	8-13
8.2.6	Unload and reload ASF definition area . . . . .	8-14
8.2.7	Run IDMSRUPD program . . . . .	8-14
	Purpose . . . . .	8-14
	Syntax . . . . .	8-14
	Parameters . . . . .	8-14
8.2.8	Execution JCL . . . . .	8-15
	OS/390 JCL . . . . .	8-15
	VSE/ESA JCL . . . . .	8-16
	VM/ESA commands . . . . .	8-16
	BS2000/OSD JCL . . . . .	8-18
8.2.9	Output from IDMSRUPD . . . . .	8-18
	Output from IDMSRUPD . . . . .	8-18
	Sample IDMSRUPD output listing . . . . .	8-18
8.2.10	Optionally, add CALC record to IDMSR schema . . . . .	8-20
8.2.11	Execution JCL . . . . .	8-20
	OS/390 JCL . . . . .	8-20
	VSE/ESA JCL . . . . .	8-21
	VM/ESA commands . . . . .	8-21
	BS2000/OSD JCL . . . . .	8-22
<b>Chapter 9. Application Environment . . . . .</b>		<b>9-1</b>
9.1	About this chapter . . . . .	9-3
9.2	Local mode processing . . . . .	9-4
9.2.1	SYSIDMS parameter file . . . . .	9-4
	What is the SYSIDMS parameter file . . . . .	9-4
	Review JCL stream for local mode programs . . . . .	9-4
	Other CA-IDMS products using SYSIDMS . . . . .	9-4
9.2.2	Security in local mode . . . . .	9-4
9.3	Deleted CA-IDMS modules . . . . .	9-5
	Remove references to deleted modules . . . . .	9-5
	SYSIDMS replaces the need for IDMSQSAM and IDMSTRAC . . . . .	9-5
9.4	New version of IDMSUTIO . . . . .	9-6
	New 12.0 version of IDMSUTIO . . . . .	9-6
9.5	COBOL (IDMSDMLC) precompiler JCL . . . . .	9-7
9.6	PL/I (IDMSDMLP) precompiler JCL . . . . .	9-8
9.7	Considerations for relinking application programs . . . . .	9-9
	Changes only necessary on the next recompile or relink . . . . .	9-9
	IDMS is the only entry module . . . . .	9-9
	Considerations by application protocol . . . . .	9-9
9.8	SPF indexes . . . . .	9-11
9.9	Region size . . . . .	9-12
9.10	CA-ADS . . . . .	9-13
	Upward compatibility . . . . .	9-13
<b>Appendix A. Dictionary Segments . . . . .</b>		<b>A-1</b>

A.1 About this appendix . . . . .	A-3
A.2 System and application dictionary segments . . . . .	A-4
<b>Appendix B. R120DMCL and R120DBTB Listing</b> . . . . .	<b>B-1</b>
B.1 About this appendix . . . . .	B-3
B.2 R120DMCL listing . . . . .	B-4
B.3 R120DBTB database name table . . . . .	B-6
<b>Appendix C. IDMSLBLS Procedure for VSE/ESA JCL</b> . . . . .	<b>C-1</b>
C.1 About this appendix . . . . .	C-3
C.2 IDMSLBLS Procedure . . . . .	C-4
What is the IDMSLBLS procedure . . . . .	C-4
IDMSLBLS procedure listing . . . . .	C-4
<b>Appendix D. External Security Managers</b> . . . . .	<b>D-1</b>
D.1 About this appendix . . . . .	D-3
D.2 Global considerations . . . . .	D-4
About this section . . . . .	D-4
D.2.1 User exits . . . . .	D-4
D.2.2 Signon processing . . . . .	D-4
Password control . . . . .	D-4
Password reverification . . . . .	D-4
Automatic terminal signon . . . . .	D-4
User session attributes . . . . .	D-5
Multiple signons . . . . .	D-5
Other considerations . . . . .	D-6
D.2.3 Resource access validation . . . . .	D-7
Protection by default . . . . .	D-7
Safe lists . . . . .	D-7
Suspending user IDs for violation threshold . . . . .	D-7
Controlling where validation will occur . . . . .	D-8
Subschema and area security . . . . .	D-8
D.3 CA-ACF2 conversion . . . . .	D-9
About this section . . . . .	D-9
D.3.1 Administration . . . . .	D-9
Specifying security options . . . . .	D-9
Resource classes . . . . .	D-9
Resource Rule Directories . . . . .	D-9
External resource names . . . . .	D-9
Using existing rules . . . . .	D-10
Environment integrity . . . . .	D-10
Defining security schemes by region . . . . .	D-11
How to define minilid . . . . .	D-11
D.3.2 Subschema security . . . . .	D-11
Using database security . . . . .	D-11
Using user exit 14 . . . . .	D-11
D.3.3 Area security . . . . .	D-12
What CA-IDMS provides . . . . .	D-12
Securing individual areas . . . . .	D-12
D.3.4 CA-ACF2 CA-IDMS interface . . . . .	D-13

D.4	CA-Top Secret conversion . . . . .	D-15
	About this section . . . . .	D-15
D.4.1	Administration . . . . .	D-15
	Resource classes . . . . .	D-15
	External resource names . . . . .	D-15
	Using existing rules . . . . .	D-15
D.4.2	Task security . . . . .	D-16
	Using external security . . . . .	D-16
	CA-Top Secret definitions . . . . .	D-16
D.4.3	Subschema security . . . . .	D-16
	Using database security . . . . .	D-16
	Using user exit 14 . . . . .	D-17
	CA-Top Secret definitions . . . . .	D-17
D.4.4	Area security . . . . .	D-18
	What CA-IDMS provides . . . . .	D-18
	Securing individual areas . . . . .	D-18
D.5	Sample user exit 14 processing . . . . .	D-19
	Subschema example . . . . .	D-19
	Area example . . . . .	D-19
<b>Appendix E. 10.2 DSECT Replacement . . . . .</b>		E-1
E.1	About this appendix . . . . .	E-3
E.2	New 12.0 macros . . . . .	E-4
	Macros replacing 10.2 COPY books . . . . .	E-4
<b>Index</b>	. . . . .	X-1

## **How to use this manual**

---

# What this document is about

This document is a guide to upgrading to a CA-IDMS Release 15.0 environment directly from a 10.x release of CA-IDMS. You must be migrating from a 10.x release of CA-IDMS to use the procedures and tools described here.

This document presents conversion guidelines for the following CA-IDMS products:

- ASF
- CA-ADS
- CA-IDMS/DB
- CA-IDMS/DC and CA-IDMS/UCF
- CA-IDMS/DDS
- The integrated data dictionary (IDD)

This document is written from the perspective of converting to CA-IDMS Release 12.0, but the same process applies to converting to CA-IDMS Release 15.0.

You can upgrade to CA-IDMS Release 15.0 from CA-IDMS Release 10.0, 10.1, 10.2, 10.21, 12.0, or 14.0. The conversion utilities provided for CA-IDMS Releases 12.0, 14.0 and 14.1 are included on the CA-IDMS Release 15.0 installation tape.

- **Upgrading from Release 10.x:** To upgrade from CA-IDMS Release 10.2 and above, perform the same conversion steps as if you were upgrading to CA-IDMS Release 12.0. Use the tools and procedures outlined in this document.

**Note:** If you are upgrading from CA-IDMS Release 10.0 or 10.1, you cannot use the DMCL Conversion Utility due to internal changes to the DMCL fields in the data dictionary in CA-IDMS Release 10.2. Instead, you must convert your DMCL syntax manually.

- **Upgrading from Release 12.0 and higher:** To upgrade from CA-IDMS Release 12.0, 14.0 or 14.1 refer to the CA-IDMS Release 14.0, 14.1 and 15.0 *Features Guide* for detailed information concerning these conversions.

**Note:** If your upgrade consists of multiple releases (i.e. 12.01 to 15), you must follow the conversion procedures documented in the *Features Guide* for the intermediate releases.

**Note:** To verify the release that is running, use the DCPROFIL or OPER tasks.

- OPER displays the CA-IDMS tape value in the upper right-hand corner as TAPE **xxxxxx**



- DCPROFIL displays the CA-IDMS and the CA-IDMS Tools, if installed, on the first DCPROFIL screen in the upper left-hand corner.

TAPE:	xxxxxx	NUMBER OF SCTS:	x
TOOLS TAPE:	xxxxxx		

For both tasks, **xxxxxx** is the tape; for example, F0GJ0B.

## **Who should use this document**

This document is for those who are involved in converting to CA-IDMS Release 12.0, 14.0, 14.1 or 15.0. Typically this includes database, data communications, or systems administrators.

Administrators should be familiar with CA-IDMS/DB and CA-IDMS/DC installation and administration and CA-IDMS Release 12.0, 14.0, 14.1 and 15.0 features.

# How this document is organized

In Chapter 1, “Introduction,” the activities involved in converting to a CA-IDMS Release 12.0 environment are presented in a table. Subsequent chapters describe these activities in the order they are listed in the table. The tasks that comprise these activities are presented as checklists, procedures, or guidelines to help you through a successful conversion to CA-IDMS Release 12.0.

This document is organized as follows:

- **Chapter 1, “Introduction”** — Presents a table of conversion activities and describes the 12.0 environment created during installation.
- **Chapter 2, “Physical Database Definition”** — Describes how to convert 10.2 DMCL definitions to 12.0 physical database definition components.
- **Chapter 3, “Security Migration”** — Describes how to convert 10.0 security definitions to 12.0 central security definitions.
- **Chapter 4, “Dictionary Migration and Setup”** — Describes how to migrate dictionaries and set up 12.0 system and application dictionaries. Also, this chapter presents options for regenerating subschemas.
- **Chapter 5, “DC/UCF System Generation and Startup”** — Identifies tasks to perform to prepare DC/UCF system definitions and startup components for use in a 12.0 environment.
- **Chapter 6, “Utilities”** — Presents a list of new utility statements and the utility programs they replace.
- **Chapter 7, “User Exits and Database Procedures”** — Identifies changes to make to user exits and database procedures for 12.0.
- **Chapter 8, “Extent Areas”** — Describes how to convert ASF extent areas to standard areas.
- **Chapter 9, “Application Environment”** — Describes changes to make to programs running in local mode.
- **Appendix A, “Dictionary Segments”** — Lists the segments included in the installed system and application dictionaries.
- **Appendix B, “R120DMCL and R120DBTB Listing”** — Provides a listing of the installed DMCL (R120DMCL) and database name table (R120DBTB).
- **Appendix C, “IDMSLBLS Procedure for VSE/ESA JCL”** — Provides a JCL procedure to be used when running CA-IDMS utilities and programs in a VSE environment.
- **Appendix D, “External Security Managers”** — Explains to users who have secured a CA-IDMS 10.2 environment with CA-ACF2 or CA-Top Secret the considerations involved in maintaining the security architecture when upgrading to CA-IDMS 12.0.

- **Appendix E, “10.2 DSECT Replacement”** — Explains the differences in the way certain CA-IDMS DSECTs are assembled under Release 12.0 compared to Release 10.2.

**Conversion aids:** CA-IDMS provides various tools to assist you during your conversion to CA-IDMS Release 12.0. These tools simplify the conversion of some of the components of your 10.2 environment. For example, CA-IDMS provides a DMCL Syntax Generator that creates 12.0 physical database definition syntax from your 10.2 DMCL definitions.

**The conversion process varies:** In many ways, the conversion process will vary based on the configuration of your environment. Because each environment is different, other conversion tasks are performed using standard CA-IDMS facilities rather than a special conversion tool. For example, journal files need to be initialized in a 12.0 environment before they are used. You use the new FORMAT utility statement to format journal files.

## Related CA documentation

Here is a list of documents you might need to reference during the conversion process:

- *CA-IDMS Features Summary (Release 12.0)*
- *CA-IDMS Features Guide — Release 14.0*
- *CA-IDMS Features Guide — Release 14.1*
- *CA-IDMS Features Guide — Release 15.0*
- CA-IDMS installation manual for your operating system
- *CA-IDMS Database Administration*
- *CA-IDMS Utilities*
- *CA-IDMS Command Facility*
- *CA-IDMS System Generation*
- *CA-IDMS System Operations*
- *CA-IDMS System Tasks and Operator Commands*
- *CA-IDMS Security Administration*
- *CA-IDMS Usage in the Siemens Environment* (BS2000/OSD users only)

## Before you begin

Before you begin the process of converting to a Release 12.0 environment, you must be familiar with the new features of CA-IDMS Release 12.0. Before you begin the conversion process:

- Read the *CA-IDMS Features Summary (Release 12.0)*
- Read these chapters in *CA-IDMS Database Administration*:
  - Physical database definition
  - Physical DDL statements
- Read sections from *CA-IDMS Security Administration* to become familiar with central security concepts
- Familiarize yourself with *CA-IDMS Utilities*
- Attend the BR120—CA-IDMS Release Enhancements course, if possible
- Read this document before performing any conversion tasks

# **Chapter 1. Introduction**

---

1.1 About this chapter . . . . .	1-3
1.2 CA-IDMS conversion activities . . . . .	1-4
1.3 Release 12.0 test environment . . . . .	1-11
1.4 Before you begin . . . . .	1-12



## 1.1 About this chapter

This chapter describes the activities involved in converting a CA-IDMS Release 10.2 environment to a Release 12.0 environment. In addition, the Release 12.0 test environment created during the installation of CA-IDMS is described.

The conversion process assumes that CA-IDMS Release 12.0 is installed in your environment.

## 1.2 CA-IDMS conversion activities

The table below lists the activities involved in converting to CA-IDMS Release 12.0. A detailed description of most conversion activities is provided in subsequent chapters of this document. There is one chapter for most major activities identified in the table. The chapters are presented in the order that the conversion activity appears in the table.

Chapter	Conversion activity	Action
Chapter 2, “Physical Database Definition”	<b>Physical database definition</b> <ul style="list-style-type: none"><li>■ Create DMCL</li><li>■ Create database name table</li></ul>	<p>Use the DMCL Syntax Generator to generate 12.0 physical database definition statements from 10.2 DMCL definition.</p> <p>Modify statements as needed and submit them to the CA-IDMS Batch Command Facility to populate the dictionary.</p> <p>Generate and punch DMCL load module.</p>
		<p>Determine requirements for 12.0 database name table.</p> <p>Define database name table by submitting appropriate statements to the CA-IDMS Batch Command Facility.</p> <p>Generate and punch database name table load module.</p>
Chapter 3, “Security Migration”	<b>Security definition migration</b> <ul style="list-style-type: none"><li>■ Migrate 10.0 security definitions to 12.0 central security definition.</li><li>■ Review 12.0 security statements.</li><li>■ Populate security system with 12.0 security definitions.</li></ul>	<ul style="list-style-type: none"><li>■ Execute security definition migration utility (RHDCSMIG).</li><li>■ Populate security system using CA-IDMS Command Facility.</li></ul>

Chapter	Conversion activity	Action
Chapter 4, “Dictionary Migration and Setup”	<b>Dictionary migration and setup</b>	<ul style="list-style-type: none"> <li>■ Area sizing</li> </ul> <p>Determine the space available in each DDLDML area you will convert.</p> <p>12.0 DDLDML area records and sets <b>do not</b> require an increase in the size of 12.0 DDLDML areas.</p>
	<p>CA-supplied internal schema and subschema definitions may require 5% additional space in the DDLDML area where they reside.</p> <ul style="list-style-type: none"> <li>■ Migrate 10.0 DDLDML areas to 12.0 format.</li> <li>■ New messages.</li> </ul>	<p>If necessary, increase the size of the DDLDML area that will contain the 12.0 CA-supplied internal schemas.</p>
	<ul style="list-style-type: none"> <li>■ CA-IDMS system task and program modules.</li> <li>■ CA-IDMS database protocol module.</li> </ul>	<p>Use the dictionary migration utility to migrate 10.0 DDLDML areas.</p> <p>Load the CA-IDMS message module from the source library provided during CA-IDMS installation into your message area.</p> <p>Load CA-IDMS system task and program modules from the source library provided during CA-IDMS installation into your system dictionary.</p> <p>Load the database protocol module from the source library provided during CA-IDMS installation into each of your migrated application dictionaries.</p>

<b>Chapter</b>	<b>Conversion activity</b>	<b>Action</b>
Chapter 4, “Dictionary Migration and Setup”	<b>Subschema regeneration</b>  Regenerate 10.0 subschemas in a 12.0 environment.	Choose from these options to generate 10.0 subschemas in 12.0 environment: <ul style="list-style-type: none"><li>■ REGENERATE ALL SUBSCHEMAS of schema syntax</li><li>■ Generate individual subschemas</li><li>■ IDMSSCON utility program</li></ul>

Chapter	Conversion activity	Action
Chapter 5, “DC/UCF System Generation and Startup”	<b>DC/UCF system generation and startup</b>	<p>■ Modify migrated system definition.</p> <p>■ Generate migrated system definition.</p> <p>■ Prepare DC/UCF environment for startup.</p> <ul style="list-style-type: none"> <li>– The format of records in these areas has changed:</li> </ul> <p style="text-align: center;">DDLCDLOG DDLCDRUN</p> <ul style="list-style-type: none"> <li>– DDLCDSCR area is different.</li> <li>– RHDCPLOG replaced by ARCHIVE LOG and PRINT LOG utility statements.</li> <li>– IDMSAJNL replaced by ARCHIVE JOURNAL utility statement.</li> <li>– Journal record formats have changed and a new journal record has been added.</li> <li>– #DCPARM macro.</li> </ul> <p>Review migrated system definition and modify as necessary. Some changes are:</p> <ul style="list-style-type: none"> <li>– Add new SYSTEM and system generation statement parameters as appropriate.</li> <li>– Review and modify as necessary, new SYSTEM and system generation statements added to your system definition during dictionary migration.</li> <li>– Include new CA-IDMS 12.0 system task and program definitions.</li> </ul> <p>Generate migrated system definition in a 12.0 environment.</p> <p>Format DDLCDLOG and DDLCDRUN areas before using them in a 12.0 environment.</p> <p>Format DDLCDSCR area.</p> <p>Prepare ARCHIVE LOG and PRINT LOG utility statements and new JCL before starting up your 12.0 system.</p> <p>Prepare ARCHIVE JOURNAL utility statement and new JCL before starting up your 12.0 system.</p> <p>Initialize journal files before using them in a 12.0 environment.</p> <p>Review #DCPARM macro parameters and make any necessary changes and then reassemble it.</p>

<b>Chapter</b>	<b>Conversion activity</b>	<b>Action</b>
Chapter 6, “Utilities”	<b>Utilities</b> <ul style="list-style-type: none"> <li>■ New utility statements replace many utility programs.</li> <li>■ New execution JCL for new utility statements.</li> <li>■ Modify 10.2 execution JCL for utility programs.</li> </ul>	<p>Prepare utility statements for each function replaced by a utility statement that you will be using.</p> <p>Backup database using BACKUP utility.</p> <p>Create execution JCL for utility statements.</p> <p>Modify 10.2 execution JCL for utility programs to identify DMCL, dictionary, and/or database name in SYSIDMS parameter file for utility programs that will run in local mode. Make other JCL changes as required.</p>
Chapter 7, “User Exits and Database Procedures”	<b>User exits and database procedures</b> <ul style="list-style-type: none"> <li>■ User exits           <ul style="list-style-type: none"> <li>– The format of control blocks has changed.</li> <li>– A user exit has been removed.</li> <li>– Four new user exits have been added.</li> </ul> </li> <li>■ Database procedures.           <p>The format of the subschema control block that is passed as a parameter to a database procedure has changed.</p> </li> </ul>	<p>Review user exits for potential changes. Modify and reassemble the code, as necessary, in a 12.0 environment.</p> <p>Replace deleted exit with new exits.</p> <p>All database procedures must be modified to use a 12.0 format subschema control block and recompiled in a 12.0 environment.</p>
Chapter 8, “Extent Areas”	<b>Extent areas and ASF</b> <p>Extent areas are no longer supported.</p>	<p>Replace queue extent areas (DDLCDQUE) with standard queue areas (DDLDCRUN).</p> <p>Convert ASF extent areas to standard areas.</p>

<b>Chapter</b>	<b>Conversion activity</b>	<b>Action</b>
None	<p><b>Reports</b></p> <ul style="list-style-type: none"> <li>■ User written DDR reports.</li> <li>■ User-modified standard DDR reports.</li> <li>■ Standard DDR reports supplied on the installation tape have been updated to include changes to 12.0 dictionary definition.</li> </ul>	Review user written reports and any standard DDR reports you have modified to determine if changes in dictionary definition require changes to reports. Make necessary changes.
Chapter 9, “Application Environment”	<p><b>Application environment</b></p> <ul style="list-style-type: none"> <li>■ New SYSIDMS parameter file.</li> <li>■ Some CA-IDMS modules have been deleted.</li> <li>■ New 12.0 version of IDMSUTIO program.</li> <li>■ New COBOL and PL/I precompiler (IDMSDMLC and IDMSDMLP) JCL.</li> <li>■ Convert SPF indexes to integrated indexes.</li> <li>■ Region size.</li> </ul>	<p>Review all execution JCL for changes. Modify execution JCL of programs that run in local mode to use the SYSIDMS parameter file.</p> <p>Re-link programs that access these programs.</p> <p>Re-link programs that access IDMSUTIO for file I/O routines.</p> <p>Review new IDMSDMLC and IDMSDMLP JCL and modify 10.2 JCL as necessary.</p> <p>Modify 10.2 schema and create 10.2 subschema replacing SPF index with an integrated index.</p> <p>Execute the 10.2 IDMSTBLU utility to convert the SPF index to an integrated index.</p> <p>If necessary, remove SPF system records from application and recompile the application.</p> <p>Review the region size of batch jobs and increase as necessary.</p>

## 1.2 CA-IDMS conversion activities

---

<b>Chapter</b>	<b>Conversion activity</b>	<b>Action</b>
Chapter 9, “Application Environment”	<b>CA-ADS</b>  CA-ADS 10.21 dialogs will run, without any modifications, in a 12.0 environment.	
None	<b>CA-OLQ</b>  Batch execution JCL	Review and modify the execution JCL for CA-OLQ batch jobs. For more information, see <i>CA-OLQ User Guide</i> .
None	<b>CA-CULPRIT</b>  Execution JCL	Review and modify the execution JCL for CULPRIT jobs. For more information, see <i>CA-Culprit User Guide</i> .
None	<b>CA-IDMS/DDS</b>  No conversion activities are required.	For information on CA-IDMS/DDS Release 12.0, see <i>CA-IDMS Features Summary (Release 12.0)</i> and <i>CA-IDMS System Generation</i> .
None	<b>CA-ICMS</b>  No changes are required to the CA-ICMS Release 12.0 environment.  CA-ICMS applications will run, without modifications, in a Release 12.0 environment.	

## 1.3 Release 12.0 test environment

During the installation process, a Release 12.0 environment is created. This environment is provided to assist you in getting started using CA-IDMS 12.0.

**What the test environment includes:** The CA-IDMS Release 12.0 test environment includes:

- Dictionary environment — Both a system and application dictionary environment are installed.  
A list of the installed segments and areas that comprise the system and application dictionary environments is in Appendix A, “Dictionary Segments.”
- DMCL module — A 12.0 DMCL called R120DMCL is installed and describes the installed 12.0 run-time environment. The R120DMCL load module is in the DBALIB load library provided during the installation process.  
You can use this DMCL to run the dictionary migration utility and to create your 12.0 DMCL and database name table modules. A description of R120DMCL is in Appendix B, “R120DMCL and R120DBTB Listing.”
- Database name table — A database name table that defines the databases and data dictionaries accessible by the R120DMCL is provided. It is called R120DBTB and a description of it appears in Appendix B, “R120DMCL and R120DBTB Listing”
- DC/UCF system — A 12.0 DC/UCF system (SYSTEM90) is installed for you to use to convert components of your 10.2 environment.
- Load libraries — Two 12.0 load libraries exist:
  - LOADLIB — includes CA-IDMS system software load modules
  - DBALIB — includes R120DMCL and R120DBTB load modules

**Review dictionary and DMCL descriptions:** You should review the list of segments and areas that comprise the installed dictionary environment and the installed DMCL to become familiar with the characteristics of your 12.0 test environment. See Appendix A, “Dictionary Segments” and Appendix B, “R120DMCL and R120DBTB Listing”

## 1.4 Before you begin

Before you begin converting any part of your CA-IDMS Release 10.2 environment, you should:

- Read the documents identified in the preface to this document
- Read the table of CA-IDMS conversion activities presented at the beginning of this chapter
- Review the information in the appendixes on the installed 12.0 test environment. You can use this environment to perform most of the conversion tasks.
- Choose a small, test 10.2 environment to convert before you convert any production environments. Once you have experience with the conversion process, conversion tools, and new CA-IDMS facilities you will be better prepared to convert your larger, production environment.
- Create a conversion plan. Identify the steps you will take to convert your initial environment. Creating a conversion plan, before you begin, will help to organize your conversion effort and keep track of conversion activities once you get started.

# Chapter 2. Physical Database Definition

---

2.1 About this chapter . . . . .	2-3
2.1.1 An overview of the process . . . . .	2-4
2.2 Convert 10.2 DMCL definitions . . . . .	2-6
2.2.1 Execute the DMCL Syntax Generator . . . . .	2-7
2.2.2 IDMSDMCC compiler batch execution JCL . . . . .	2-9
2.2.2.1 OS/390 execution JCL . . . . .	2-9
2.2.2.2 VSE/ESA execution JCL . . . . .	2-10
2.2.2.3 VM/ESA commands . . . . .	2-11
2.2.2.4 BS2000/OSD execution JCL . . . . .	2-11
2.2.3 What the DMCL Syntax Generator produces . . . . .	2-12
2.2.3.1 DMCL statements . . . . .	2-14
2.2.3.2 Segment definitions within the DMCL . . . . .	2-16
2.2.4 Correct errors from IDMSDMCC . . . . .	2-18
2.3 Create 12.0 DMCL module . . . . .	2-19
2.4 Create a 12.0 database name table . . . . .	2-23
2.4.1 Specifying the default dictionary . . . . .	2-23
2.4.2 Grouping segments . . . . .	2-24
2.4.3 Specifying a DBNAME at run time . . . . .	2-25
2.4.4 Guidelines for defining database name tables . . . . .	2-25
2.4.5 Basic DBTABLE definition . . . . .	2-25
2.4.6 Secondary dictionaries . . . . .	2-27
2.4.7 Conflicting area names . . . . .	2-27
2.4.8 Non-dictionary DBNAMEs . . . . .	2-28
2.4.9 Mixed page groups . . . . .	2-29



## 2.1 About this chapter

This chapter describes how to process components of your Release 10.2 database environment to create a Release 12.0 physical database definition.

**Convert DMCL before migrating dictionary:** You must process your 10.2 DMCL definition(s) from a 10.2 DDLDML dictionary area before migrating the DDLDML area.

**Conversion tools and environment:** To generate a 12.0 physical database definition from a 10.2 DMCL definition you need:

This component	For these tasks
10.2 DMCL Compiler	To modify or validate 10.2 DMCLs, if necessary, before submitting them for processing by the DMCL Syntax Generator.
Release 10.2 data dictionary	The DMCL Syntax Generator extracts definitions from a 10.2 data dictionary.
Release 10.2 run-time environment <ul style="list-style-type: none"> <li data-bbox="502 1051 1019 1083">■ Load libraries</li> <li data-bbox="502 1100 1019 1132">■ DMCL</li> <li data-bbox="502 1148 1019 1248">■ IDMSNWKA subschema describing the dictionary containing the DMCL to be converted</li> <li data-bbox="502 1267 1019 1366">■ For local mode processing, you need the IDMSDBTB database name table for processing secondary dictionaries</li> </ul>	To run the DMCL Syntax Generator in a 10.2 environment.
DMCL Syntax Generator The DMCL Syntax Generator (IDMSDMCC program) is a 10.2 module that must exist in a stand-alone load library and can only run in a 10.2 environment. It should have been installed in a separate load library during the installation process.	To process 10.2 DMCL definitions and generate 12.0 physical database definition statements.

This component	For these tasks
Release 12.0 run-time environment <ul style="list-style-type: none"><li>■ Load libraries</li><li>■ DMCL</li><li>■ Data dictionary (database catalog segment which includes DDLCAT, DDLCATX, and DDLCATLOD areas)</li></ul>	To define, generate, punch, and link-edit physical database components.
CA-IDMS Batch Command Facility	<p>To process 12.0 physical database definition statements and populate the database catalog segment of a 12.0 data dictionary.</p> <p>To generate and punch 12.0 DMCL and database name table modules.</p>

**Related CA documentation:** You may need to refer to these documents when modifying 12.0 physical database definition statements produced by the DMCL Syntax Generator.

- *CA-IDMS Database Administration*
- *CA-IDMS Utilities*
- *CA-IDMS Command Facility*

### 2.1.1 An overview of the process

To create and generate Release 12.0 physical database definition components, follow these steps:

Step	Action
Create 12.0 physical database definition statements from 10.2 DMCL definitions	<ol style="list-style-type: none"><li>1. Identify the 10.2 DMCLs to convert to 12.0</li><li>2. Ensure the 10.2 DMCLs are validated</li><li>3. Code input statements to the DMCL Syntax Generator to identify the DMCL you want to convert</li><li>4. Process 10.2 DMLCs by executing IDMSDMCC against the 10.2 data dictionary that contains the DMCL</li><li>5. Review the transaction summary supplied by the DMCL Syntax Generator to verify the DMCL was successfully processed</li></ol>

Step	Action
Create 12.0 DMCL module	<ol style="list-style-type: none"><li>1. Review and modify the physical database definition statements created by the DMCL Syntax Generator</li><li>2. Populate the database catalog segment of your system dictionary by submitting modified physical database definition statements to the CA-IDMS Batch Command Facility</li><li>3. Generate, punch, and link-edit 12.0 DMCL load module</li></ol>
Create 12.0 database name table	<ol style="list-style-type: none"><li>1. Create CREATE DBTABLE and DBNAME statements</li><li>2. Submit statements to the CA-IDMS Batch Command Facility</li><li>3. Generate, punch, and link edit 12.0 database name table module</li></ol>

**What's next:** The remainder of this section describes how to perform each of the steps identified in the above table.

## 2.2 Convert 10.2 DMCL definitions

**12.0 physical database components** In Release 12.0, the process of defining a physical database is completely separate from defining the logical description of a database. Defining a physical database includes defining::

- Segments — segment definitions include area and file definitions as well as a page group assignment and the maximum number of records or rows per page.
- DMCL — A DMCL definition includes the run-time description of a physical database environment. It includes segment, journal and buffer definitions.

**The DMCL Syntax Generator:** The DMCL Syntax Generator (IDMSDMCC) generates 12.0 physical database definition statements from validated 10.2 DMCL and schema definitions. The DMCL Syntax Generator executes as a stand-alone batch compiler and will only process DMCL definitions that have been generated with the Release 10.2 DMCL compiler.

You identify the 10.2 DMCLs you want to process by submitting input parameters to IDMSDMCC. Since an area can be included in multiple DMCLs, consider choosing a global DMCL for syntax conversion.

**Sequence of statements:** The DMCL Syntax Generator creates

- One CREATE SEGMENT statement for each unique schema name and version associated with the areas included in the 10.2 DMCL.
- CREATE FILE statements for those files associated with the AREA statements included in the 10.2 DMCL.
- CREATE AREA statements for those areas included in the 10.2 DMCL.

**CAUTION:**

**The DMCL Syntax Generator only converts 10.2 area and file definitions for those areas and files included in the 10.2 DMCL you are converting. You must explicitly add area and file definitions to your 12.0 physical database definition that are part of a schema but are *not* included in the 10.2 DMCL you are converting.**

- DMCL statements following SEGMENT, FILE, and AREA statements.

**Before you run the DMCL Syntax Generator:** Before you run the DMCL Syntax Generator, review these items:

1. Ensure that 10.2 DMCL definitions are validated.
2. The DMCL Syntax Generator runs in a 10.2 environment in either local mode or under the central version.
3. While the DMCL Syntax Generator can process multiple DMCLs in a single job, you should consider processing one DMCL at a time.
4. Do not include any 12.0 load libraries in the IDMSDMCC execution JCL.

## 2.2.1 Execute the DMCL Syntax Generator

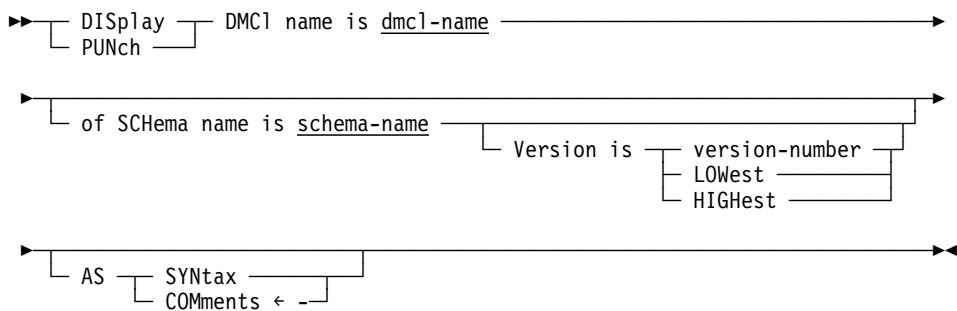
**Purpose:** The DMCL Syntax Generator processes 10.2 DMCL definitions and generates appropriate Release 12.0 physical database definition syntax.

The DMCL Syntax Generator is a data dictionary compiler and supports the IDD SIGNON statement and the CA-IDMS DDL compiler SET OPTIONS statement.

- For a complete description of the IDD SIGNON statement, see *CA-IDMS IDD DDDL Reference Guide*
- For a complete description of the CA-IDMS DDL Compiler SET OPTIONS statement, see *CA-IDMS Database Administration*.

**Authorization:** To display or punch DMCL definitions from a Release 10.2 dictionary, you must have authority to use the 10.2 DMCL compiler.

If you have dictionary signon security turned on or you are accessing a secondary dictionary and need to specify a DICTNAME, you must include a SIGNON DICTIONARY statement in the SYSIPT file before any DMCL Syntax Generator input statements.

**Syntax****Parameters****DISplay**

Indicates that IDMSDMCC will display the physical database definition syntax it will generate.

**PUNch**

Indicates that IDMSDMCC will display and punch the physical database definition syntax it will generate.

**DMCL name is dmcl-name**

Identifies the name of the Release 10.2 DMCL definition for which 12.0 syntax is generated.

**of SCHEMA name is schema-name**

Identifies the name of the schema associated with the named DMCL.

*Schema-name* is required only if the DMCL name is not unique in the dictionary.

**Version is**

Optionally qualifies the named schema with a version number.

**Version-number**

Specifies the explicit version number assigned to the named schema.

*Version-n* is in the range 1 through 9999 and defaults to the current session option.

**LOWest**

Specifies the lowest version number assigned to the named schema.

**HIGHest**

Specifies the highest version number assigned to the named schema.

**AS SYNTAX**

Indicates that IDMSDMCC will produce the physical database definition statements as syntax which you can then edit and resubmit to the CA-IDMS Batch Command Facility.

**COMments**

Indicates that IDMSDMCC will produce the physical database definition statements with comment identifiers in columns 1 and 2.

## Usage

*Converting multiple DMCLs with the same name:* If you have multiple 10.2 DMCLs with the same name, the DMCL Syntax Generator will generate multiple CREATE DMCL statements with the same name. You should resolve multiple DMCL names. Here are some suggestions:

- Change the name of the DMCL or
- Delete the DMCL if you don't need it or
- Assign the segments and other physical database components to another DMCL

## 2.2.2 IDMSDMCC compiler batch execution JCL

You submit the DMCL syntax generator statements to the IDMSDMCC compiler. IDMSDMCC will run in either local mode or under the central version.

### 2.2.2.1 OS/390 execution JCL

Here is the OS/390 execution JCL when running IDMSDMCC under the central version.

**CAUTION:**

**Do not include any Release 12.0 load libraries in the execution JCL for IDMSDMCC.**

#### IDMSDMCC JCL

```
//IDMSDMCC EXEC PGM=IDMSDMCC,REGION
//STEPLIB DD DSN=dmcc.loadlib,DISP=SHR
//          DD DSN=cdms.loadlib,DISP=SHR
//sysctl  DD DSN=dbdc.sysctl,DISP=SHR
//SYSLST  DD SYSOUT=A
//SYSPCH  DD SYSOUT=B
//SYSIPT  DD *
  
Include dictionary SIGNON statement here
Include IDMSDMCC input statements here
```

<u>dmcc.loadlib</u>	Data set name of stand-alone load library that contains the IDMSDMCC program
<u>cdms.loadlib</u>	Data set name of 10.2 system software load library
<u>sysctl</u>	DDname for SYSCTL file
<u>dbdc.sysctl</u>	Data set name of SYSCTL file

To execute the DMCL Syntax Generator in local mode:

- Add the name of the load library that contains your 10.2 run-time DMCL and network subschema load modules. Also, include the IDMSDBTB database name table if you are using a non-default dictionary.
- Add the name of the data dictionary DDLDML area containing the DMCL you are processing.
- Add the appropriate journal specifications.

### 2.2.2.2 VSE/ESA execution JCL

Here is the VSE/ESA execution JCL when running IDMSDMCC under the central version.

**CAUTION:**

**Do not include any Release 12.0 load libraries in the execution JCL for IDMSDMCC.**

#### IDMSDMCC JCL

```
// LIBDEF *,SEARCH=10.2 libraries
// DLBL dmcc,'cdms.dmcc.lib'
// ASSGN SYSxxx,DISK,VOL=nnnnnn,SHR
// DLBL IDSPCH,'syspch.dsn'
// EXTENT SYSPCH,nnnnnn,,ssss,1111
// EXEC IDMSDMCC,SIZE=1048K
Include dictionary SIGNON statement of SIGNON SECURITY is on
/*
Include IDMSDMCC input statements here
/*
```

<u>10.2 libraries</u>	10.2 library definitions
<u>cdms.dmcc.lib</u>	File identifier of the library containing the IDMSDMCC program
<u>syspch.dsn</u>	Filename of the SYSPCH file

To execute the DMCL Syntax Generator in local mode:

- Add the name of the load library that contains your 10.2 run-time DMCL and network subschema load modules. Also, include the IDMSDBTB database name table if you are using a non-default dictionary.
- Add the name of the data dictionary DDLDML area containing the DMCL you are processing.
- Add the appropriate journal specifications.

### 2.2.2.3 VM/ESA commands

Here are the VM/ESA commands to run IDMSDMCC under the central version.

**CAUTION:**

**Do not include any Release 12.0 load libraries in the execution JCL for IDMSDMCC.**

**IDMSDMCC JCL**

```
FILEDEF SYSLST PRINTER
FILEDEF SYSRPT PRINTER
FILEDEF SYSPCH PRINTER
FILEDEF SYSIPT DISK idmsdmcc input a
GLOBAL LOADLIB dmcc idmslib
OSRUN IDMSDMCC
```

---

<u>dmcc</u>	Filename of the load library containing the IDMSDMCC program
<u>idmslib</u>	Filename of the CA-IDMS 10.2 load library

---

To execute the DMCL Syntax Generator in local mode:

- Add the name of the load library that contains your 10.2 run-time DMCL and network subschema load modules. Also, include the IDMSDBTB database name table if you are using a non-default dictionary.
- Add the name of the data dictionary DDLDML area containing the DMCL you are processing.
- Add the appropriate journal specifications.

### 2.2.2.4 BS2000/OSD execution JCL

Here is the BS2000/OSD execution JCL when running IDMSDMCC under the central version.

**CAUTION:**

**Do not include any Release 12.0 load libraries in the execution JCL for IDMSDMCC.**

**IDMSDMCC JCL**

```
/ FILE      LINK=sysctl,dbdc.sysctl,SHARUPD=YES  
/ FILE      LINK=CDMSLIB,cdms.loadlib  
/ SYSFILE   SYSOPT=cdms.sysopt  
/ SYSFILE   SYSDTA=(SYSCMD)  
/ EXEC      (IDMSDMCC,dmcc.loadlib)  
Include dictionary SIGNON statements here  
Include IDMSDMCC input statements here
```

<u>dbdc.sysctl</u>	Filename of SYSTCL file
<u>cdms.loadlib</u>	Filename of 10.2 system software load library
<u>cdms.sysopt</u>	Filename of the SYSOPT file
<u>dmcc.loadlib</u>	Filename name of stand-alone load library that contains the IDMSDMCC program

To execute the DMCL Syntax Generator in local mode:

- Add the name of the load library that contains your 10.2 run-time DMCL and network subschema load modules. Also, include the IDMSDBTB database name table if you are using a non-default dictionary.
- Add the name of the data dictionary DDLDML area containing the DMCL you are processing
- Add the appropriate journal specifications

### 2.2.3 What the DMCL Syntax Generator produces

**CREATE SEGMENT statement:** A segment statement is created for each unique schema identified in the OF SCHEMA NAME clause of the AREA statements of the named DMCL.

The PAGE GROUP and the MAXIMUM RECORDS PER PAGE (formerly the PAGE CONTAINS clause of the SCHEMA statement) are also part of the CREATE SEGMENT statement

The DMCL Syntax Generator creates segment names as follows:

- The segment name is the same as the schema name for the first *schema-name version-n* encountered in the DMCL
- Subsequent occurrence(s) of the same schema name with different version numbers will be assigned segment names in the form:

SEGnnnn starting with SEG00001 through SEG99999.

**Sample output:** This is the name of a segment and the name and version of the schema to which the segment applies.

This segment name is created from another occurrence of the same schema name with a different version number.

```
IDMSDMCC 10.2      COMPUTER ASSOCIATES INTERNATIONAL, INC.      DATE      PAGE
S10200          10.2 DMCL TO 12.0 SYNTAX GENERATOR           10/31/90  0001

0000001      DISPLAY DMCL TESTDMCL AS SYNTAX.
**+ =====
**+ SEGMENT TESTSCHM CREATED FROM SCHEMA TESTSCHM VERSION 1 ;
CREATE SEGMENT TESTSCHM
    PAGE GROUP 0
    MAXIMUM RECORDS PER PAGE 255
;
**+ SEGMENT SEG00001 CREATED FROM SCHEMA TESTSCHM VERSION 2 ;
CREATE SEGMENT SEG00001
    PAGE GROUP 0
    MAXIMUM RECORDS PER PAGE 255
;
```

**CREATE FILE statement:** A CREATE FILE statement is created for each file included in the 10.2 DMCL.

The ASSIGN TO clause will reflect the external name associated with the DMCL file name either through the default schema file definition or through a DMCL file override.

**Native VSAM:** An access method clause is generated for native VSAM files only.

If the file is a native VSAM KSDS or PATH file, you must modify the CREATE FILE statement to include appropriate FOR CALC and/or FOR SET clauses on the access method clause.

#### *Sample output*

```
CREATE FILE TESTSCHM.MULTIPLE-FILE-1
    ASSIGN TO MULFILE1
;
CREATE FILE TESTSCHM.MULTIPLE-FILE-2
    ASSIGN TO MULTFIL2
;
CREATE FILE TESTSCHM.MULTIPLE-FILE-3
    ASSIGN TO MULFILE3
;
```

**CREATE AREA statement:** A CREATE AREA statement is created for each area included in the 10.2 DMCL.

The value specified in the PRIMARY SPACE clause is the value specified on the PAGE RANGE IS clause of the SCHEMA AREA statement.

PAGE SIZE and ORIGINAL PAGE SIZE information is extracted directly from the 10.2 DMCL AREA statement PAGE CONTAINS and SPACE MANAGEMENT INTERVAL clauses.

Symbolics are not generated.

Alias area names are not used by the IDMSDMCC compiler. IDMSDMCC uses the area name specified in the ALIAS OF *area-name* OF SCHEMA *schema-name* clause as the 12.0 physical area name.

**Native VSAM:** A WITHIN FILE clause will be generated for each non-path file mapping in the DMCL area. If the file map is a native VSAM ESDS, KSDS, or RRDS file, no FROM/THRU subclause is generated. If the file map is a native VSAM path file, a WITHIN PATH FILE clause is generated in place of the WITHIN FILE clause.

**Sample output:** PRIMARY SPACE is created from the SCHEMA AREA statement PAGE RANGE IS clause.

PAGE SIZE and ORIGINAL PAGE SIZE are created from the DMCL AREA statement PAGE CONTAINS and SMI BASED ON clauses.

```
CREATE PHYSICAL AREA TESTSCHM.MULT-FILE-AREA  
PRIMARY SPACE 100 PAGES FROM PAGE 400300  
PAGE SIZE 6144 CHARACTERS  
ORIGINAL PAGE SIZE 5120 CHARACTERS  
WITHIN FILE MULTIPLE-FILE-1 FROM 1 THRU 25  
WITHIN FILE MULTIPLE-FILE-2 FROM 1 THRU 30  
WITHIN FILE MULTIPLE-FILE-3 FROM 1 THRU 45  
;
```

### 2.2.3.1 DMCL statements

Once the segment statements are created, the DMCL Syntax Generator builds these release 12.0 DMCL statements:

- CREATE DMCL statement
- CREATE BUFFER statements
- CREATE JOURNAL BUFFER statement
- CREATE JOURNAL statements

**CREATE DMCL statement** A CREATE DMCL statement is created for the 10.2 DMCL you are processing. The name of the 10.2 DMCL becomes the name of the Release 12.0 DMCL.:

**Sample output**

```
CREATE DMCL TESTDMCL ;
```

**CREATE BUFFER statement:** A buffer statement is created for each standard buffer defined in the 10.2 DMCL.

PAGE SIZE is the value specified on the PAGE CONTAINS clause in the 10.2 DMCL.

LOCAL MODE and CV MODE BUFFER INITIAL PAGES and MAXIMUM PAGES clauses are based on the 10.2 DMCL BUFFER CONTAINS clause. You may want to modify these values to create different size buffer pools for local mode and CV operations.

**Note:** MAXIMUM PAGES clause is not generated for native VSAM buffers.

*Native VSAM:* For native VSAM buffers, either of the following clauses is generated:

- NATIVE VSAM LSR KEYLEN *key-length* STRNO *string-number*

or

- NATIVE VSAM NSR BUFNI *buffer-number* STRNO *string-number*

*Sample output:* PAGE SIZE is created from the PAGE CONTAINS clause on the BUFFER statement in the 10.2 DMCL.

LOCAL MODE and CV MODE BUFFER are created from the BUFFER CONTAINS clause of the BUFFER statement in the 10.2 DMCL.

```
CREATE BUFFER TESTDMCL.MULTIPLE-FILES
  PAGE SIZE 6144 CHARACTERS
    LOCAL MODE BUFFER PAGES 9
    CV MODE BUFFER INITIAL PAGES 9 MAXIMUM PAGES 9
;
CREATE BUFFER TESTDMCL.MULTIPLE-TEST
  PAGE SIZE 6144 CHARACTERS
    LOCAL MODE BUFFER PAGES 9
    CV MODE BUFFER INITIAL PAGES 9 MAXIMUM PAGES 9
;
CREATE BUFFER TESTDMCL.BUF1-TEST
  PAGE SIZE 3156 CHARACTERS
  LOCAL MODE BUFFER PAGES 4
  CV MODE BUFFER INITIAL PAGES 4 MAXIMUM PAGES 4
;
```

**CREATE JOURNAL BUFFER statement:** A CREATE JOURNAL BUFFER statement is created for the journal buffer defined in the 10.2 DMCL. If no journal buffer is defined in the 10.2 DMCL, you must define a journal buffer in the 12.0 DMCL before you generate it.

*Sample output*

```
CREATE JOURNAL BUFFER TESTDMCL.JOURNAL-BUFFER  
PAGE SIZE 6144 CHARACTERS  
BUFFER PAGES 9  
;
```

**CREATE JOURNAL statements:** The appropriate 12.0 DMCL journal statement is created for each journal file defined in the 10.2 DMCL. There are three types of JOURNAL statements:

- DISK JOURNAL
- ARCHIVE JOURNAL
- TAPE JOURNAL

**CREATE DISK JOURNAL statement:** A CREATE DISK JOURNAL statement is generated for each DISK JOURNAL FILE statement in the 10.2 DMCL.

*Sample output*

```
CREATE DISK JOURNAL TESTDMCL.DISK-JOURNAL  
FILE SIZE 5000 BLOCKS  
ASSIGN TO SYSJRNL  
;
```

**CREATE ARCHIVE JOURNAL statement:** A CREATE ARCHIVE JOURNAL statement is created for each ARCHIVE JOURNAL statement in the 10.2 DMCL.

*Sample output*

```
CREATE ARCHIVE JOURNAL TESTDMCL.ARCHIVE-JOURNAL  
BLOCK SIZE 4096 CHARACTERS  
ASSIGN TO SYSARCH  
;
```

**CREATE TAPE JOURNAL:** A CREATE TAPE JOURNAL statement is created for each TAPE JOURNAL statement if one is included in the 10.2 DMCL.

### 2.2.3.2 Segment definitions within the DMCL

**ALTER DMCL statement:** Once the DMCL is defined, segment definitions and other components of the physical database definition are associated with it. To complete the 12.0 DMCL definition, the DMCL Syntax Generator creates:

- ALTER DMCL statement
- DEFAULT BUFFER clause specifying the first standard buffer defined in the 10.2 DMCL
- INCLUDE SEGMENT clauses to associate the previously defined segments with the DMCL
- INCLUDE FILE clauses for each file in the segment to explicitly assign the file to a buffer

**Note:** INCLUDE FILE statements are not generated for files assigned to the DEFAULT BUFFER.

- INCLUDE AREA clauses as required for each area in the segment to accommodate the PAGE RESERVE clause if it was included in the 10.2 DMCL

*Sample output:* The INCLUDE SEGMENT clause associates the TESTSCHM segment with the TESTDMCL DMCL.

This INCLUDE FILE clause identifies the buffer the file uses.

This INCLUDE AREA clause identifies a page reserve for the MULT-FILE-AREA area.

```
ALTER DMCL TESTDMCL
    DEFAULT BUFFER DEFAULT-BUFFER
    INCLUDE SEGMENT TESTSCHM
        INCLUDE FILE TESTSCHM.MULTIPLE-FILE-1
            BUFFER MULTIPLE-FILES
        INCLUDE FILE TESTSCHM.MULTIPLE-FILE-2
            BUFFER MULTIPLE-FILES
        INCLUDE FILE TESTSCHM.MULTIPLE-FILE-3
            BUFFER MULTIPLE-FILES
        INCLUDE PHYSICAL AREA TESTSCHM.MULT-FILE-AREA
            PAGE RESERVE 200
    INCLUDE SEGMENT SEG00001
        INCLUDE FILE SEG00001.MULTIPLE-TEST-1
            BUFFER MULTIPLE-TEST
        INCLUDE FILE SEG00001.MULTIPLE-TEST-2
            BUFFER MULTIPLE-TEST
        INCLUDE FILE SEG00001.MULTIPLE-TEST-3
            BUFFER MULTIPLE-TEST
    INCLUDE PHYSICAL AREA SEG00001.MULT-FILE-AREA
        PAGE RESERVE 120
;
```

**Transaction summary:** The transaction summary summarizes the results of the processing by the DMCL Syntax Generator.

Error messages are displayed before the transaction summary.

*Sample transaction summary*

ENTITY	TRANSACTION SUMMARY				ADD MODIFY REPLACE DELETE DISPLAY
	.....	.....	.....	.....	
DMCL	0	0	0	0	1
NO ERRORS OR WARNINGS ISSUED FOR THIS COMPILE					

#### 2.2.4 Correct errors from IDMSDMCC

The possible errors you might receive from IDMSDMCC are those that the 10.2 DMCL compiler produces with one addition. If the DMCL you are converting isn't in a 10.2 format or isn't validated, IDMSDMCC generates the following message:

E DC643253 DMCL NOT 10.2 FORMAT OR CONTAINS ERRORS - 10.2  
VALIDATE REQUIRED

If you receive this message, check to see if the DMCL is validated using the 10.2 DMCL compiler. Validate the DMCL if it is not validated.

## 2.3 Create 12.0 DMCL module

**Review the output from the IDMSDMCC compiler:** You should review the physical database definition statements the IDMSDMCC compiler generates and modify them to reflect the requirements of your 12.0 database environment.

**What to review:** Here is a list of some of the items you should review before defining, generating, and punching a 12.0 DMCL. This is not meant to be a complete list and you should make all the necessary changes to meet the needs of your environment.

After you modify the physical database definition statements, submit them to the CA-IDMS Batch Command Facility for processing.

- For a complete description of the Batch Command Facility, see *CA-IDMS Command Facility*.

Action	Statement
Add area and file definitions for those areas and files included in a schema and not included in the converted 10.2 DMCL. You must add these definitions to ensure there are no missing page and file block ranges.	CREATE FILE and AREA statements.
Review segment names and change them if they don't conform to your naming conventions.  ►► It is recommended that you create a system dictionary that is separate from application dictionaries. Before you generate and punch your global DMCL module, see Chapter 4, "Dictionary Migration and Setup" for a discussion of setting up a system dictionary.	CREATE SEGMENT, FILE, and AREA statements and INCLUDE SEGMENT, FILE, and AREA clauses under the ALTER DMCL statement.

---

Action	Statement
Review area names and segment assignments.	CREATE AREA statement and possibly the INCLUDE AREA and FILE clauses.
<ul style="list-style-type: none"><li>■ Change area names for dictionary DML and load areas to DDLDML and DDLCLOD. If you have multiple, DDLDML and DDLCLOD areas, assign them to different segments.</li><li>■ Review the names of the areas in your subschemas and verify that they match area names in your segments. If they are different, change the area names in your segment definitions.</li><li>■ Assign user database and dictionary areas to segments to support your processing requirements.</li></ul>	CREATE AREA and possibly the INCLUDE AREA clause.
Assign the message area (DDLCMSG) to the SYSMSG segment.	CREATE AREA and FILE statements and INCLUDE SEGMENT AREA and FILE clauses.
Review the R120DMCL and determine if you need to define any of its dictionaries and sample databases in your global DMCL.  ►►Additional segment definitions are required if you are using the CA-IDMS central security facility. For more information on using the CA-IDMS central security facility, refer to Chapter 3, “Security Migration.”	AREA statement and INCLUDE SEGMENT and AREA clauses.  Add the appropriate statements to your global DMCL definition.

---

Action	Statement
Add database name table name to DMCL definition.	DBTABLE clause of the ALTER DMCL statement.
While CA-IDMS does not require you to define a database name table, you will need to have a database name table to define a default dictionary, define database names and associate segments with them and to accommodate other run-time functions.	
You can add a database name table name to the DMCL definition before you actually define and punch the database name table. However, at run-time both the DMCL load module and the DBTABLE module it references must exist.	
Review buffer specifications and the new BUFFER statement parameters. For example, local mode and central version buffer pool sizes can be different in the same DMCL.	Modify the BUFFER statement to alter parameters or include new parameters.
Ensure that a journal buffer is defined.	CREATE JOURNAL BUFFER.
Native VSAM users must modify FILE statements.	Change the FILE statement.
If the file is a native VSAM file or a native VSAM file representing a VSAM data set and index on which a set is defined, modify the FILE statement to include appropriate FOR CALC and/or FOR SET clauses on the access method clause.	
Define the physical database definition and populate the database catalog segment of the data dictionary using the CA-IDMS Batch Command Facility.	
Generate and punch the DMCL load module.	GENERATE DMCL. PUNCH DMCL LOAD MODULE utility statement.

## 2.3 Create 12.0 DMCL module

---

Action	Statement
Link-edit the resulting object module to a load (core-image) library.	Use the linkage-editor for your operating system.

## 2.4 Create a 12.0 database name table

CA-IDMS/DB requires the presence of a database name table for most processing. Typically you only need one database name table for all DMCLs defined within a system dictionary. You do not need to create separate database name tables for each DMCL.

**Why you need a database name table:** You need a database name table:

1. To define a default dictionary for both online and local mode processing
2. To group multiple segments under one name for processing as a single database
3. To identify the database to be accessed by a run unit when no database name is provided by the application or run-time environment

**What's next:** The remainder of this chapter describes the differences between 10.0 and 12.0 database name tables and then provides guidelines for defining a database name table in Release 12.0 based on these differences.

►►For a complete discussion of database name tables, refer to *CA-IDMS Database Administration*.

### 2.4.1 Specifying the default dictionary

**What is the default dictionary:** The default dictionary is the dictionary accessed whenever one is not specified. In Release 10.2, it was sometimes referred to as the primary dictionary and its areas were those whose page ranges were in the IDMSNWKA subschema.

In Release 12.0, page ranges are no longer in subschemas; therefore another mechanism must be used to identify the default dictionary. The database name table is that mechanism.

**Default dictionary identified as a DBNAME:** The default dictionary is identified as the DBNAME specified in the DBTABLE mapping for the IDMSNWKL subschema. Since you typically want to map all subschemas whose names begin with "IDMSNWK" in the same way, the DBTABLE mapping statement usually looks like this:

```
SUBSCHEMA IDMSNWK? MAPS TO IDMSNWK? DBNAME SYSDICT
```

Every database name table must have a default dictionary specification.

## 2.4.2 Grouping segments

When binding a run unit in 10.2, the page ranges in the subschema identified the areas to be accessed. In 12.0, subschemas no longer contain page ranges so the areas to be accessed by a run unit must be identified in some other way.

**Segment name as DBNAME:** If the areas to be accessed are all within one segment, the application can simply specify the name of the segment as the DBNAME on the BIND RUNUNIT statement (or specify it externally through DCUF or SYSIDMS parameters).

For example, the EMPLOAD program executed during CA-IDMS installation only requires access to areas of the EMPDEMO segment. The SYSIDMS parameter file in the execution job stream specifies DBNAME=EMPDEMO, identifying the segment to be accessed. No special DBNAME entry is required.

**Multiple segments associated with a DBNAME:** If, on the other hand, the application must access areas within multiple segments, those segments must be identified as part of the definition of a DBNAME in the database name table. When the bind takes place, CA-IDMS/DB searches the segments associated with the DBNAME for a match on the area name in the subschema. Only one area will match because the name of all areas in segments associated with a DBNAME must be unique. If this condition is not satisfied, the DBNAME is flagged as "in error" and run units receive a 1494 error status when binding to the DBNAME. To avoid this situation, use the DMCL option of the IDMSLOOK utility to validate your database names prior to making the DMCL or DBTABLE available for processing.

**Examples:** The DBTABLE provided during installation again provides an example of using a DBNAME to group segments together as one database.

The system dictionary is the dictionary used to contain both physical database definitions (DMCLs, SEGMENTs, etc.) and the DC/UCF system definition. The logical name of this dictionary must be SYSTEM, since components of the run-time system access it under this name. However, it is composed of multiple segments:

- The CATSYS segment containing the DDLCAT, DDLCATX and DDLCATLOD areas
- The SYSTEM segment containing the DDLDML, DDLDCLOD and other system run-time areas
- The SYMSG segment containing the messages issued by the run-time system

In order to treat all of these segments as a single database for processing by tools such as IDD and the CA-IDMS Command Facility, the database name table contains a DBNAME called SYSTEM which includes all three segments as part of its definition.

### 2.4.3 Specifying a DBNAME at run time

In 10.2, it was possible to access a database without naming it. The only requirement was that the application identify a subschema whose page ranges matched those of the areas to be accessed. In 12.0, because subschemas no longer contain page ranges, the name of the database or segment to be accessed must be available at run time.

**Options for specifying DBNAME at run time:** A database name or segment name can be specified at run time any of the following ways:

- By the application, using the DBNAME parameter on the BIND RUNUNIT statement.
- From the session profile DBNAME attribute. Session attributes are established through user or system profiles, DCUF SET commands in DC/UCF or SYSIDMS parameters in batch.
- From the database name table through the use of subschema mapping parameters on the DBTABLE statement.

**No DBNAME specified during bind processing:** When binding a run unit in 12.0, if no DBNAME has been established by the application or the session profile, CA-IDMS/DB searches the list of subschema mappings looking for one in which the *from-subschema* matches the name of the subschema specified on the BIND statement. If a match is found, the DBNAME specified in the subschema mapping identifies the segments (and areas) to be accessed by the run unit. If no match is found, the bind operation fails with an error status of 1491.

To ensure that run units will bind successfully, you must specify subschema mappings for all run units that bind without establishing a DBNAME through application program logic or session profile settings.

### 2.4.4 Guidelines for defining database name tables

It is impossible to provide a complete set of rules for defining database name tables, since each site will have different considerations. However, it is possible to provide some examples of simple situations and some points to be considered in more complex cases. Before defining your database name table, you should also read the appropriate sections in *CA-IDMS Database Administration*.

You define 12.0 database name tables by submitting statements to the CA-IDMS Batch Command Facility.

### 2.4.5 Basic DBTABLE definition

If your 10.2 system has no database names defined or has no duplicate area names (you did not have to use the ALIAS parameter of the AREA statement in defining your 10.2 DMCL), then define your 12.0 database name table as follows:

1. Define a DBNAME for each of your dictionaries. You will have a DBNAME for:

- Your one and only application dictionary which includes the segment containing the DDLDML and DDLDCLOD areas of your migrated 10.2 dictionary and the SYSSMSG segment containing your 10.2 DDLDCLMSG area populated with 12.0 CA-IDMS messages
  - The SYSTEM dictionary which should include the same segments as those defined during installation
  - The SYSDIRL dictionary as defined during installation, unless you run IDMSDIRL against your application dictionary
2. On the DBTABLE statement, identify the application dictionary as your default dictionary by including the following:
- ```
SUBSCHEMA IDMSNWK? MAPS TO IDMSNWK? DBNAME DEFdict
```
- Where DEFdict is the DBNAME of your application dictionary
3. Define a default DBNAME that includes all non-dictionary segments
4. As the last subschema mapping parameter on the DBTABLE statement, include the following:
- ```
SUBSCHEMA ???????? MAPS TO ???????? DBNAME DEFdb
```
- Where DEFdb is the name given to the default DBNAME.

This will allow your applications to issue BIND RUNUNIT statements, even though they do not specify a DBNAME, since all subschemas (except those beginning with IDMSNWK) will use the default DBNAME.

The complete database name table might look like this:

```
CREATE DBTABLE ALLDBS  
SUBSCHEMA IDMSNWK? MAPS TO IDMSNWK? DBNAME DEFdict  
SUBSCHEMA ???????? MAPS TO ???????? DBNAME DEFdb;  
  
CREATE DBNAME ALLDBS.SYSTEM  
SEGMENT CATSYS  
SEGMENT SYSTEM  
SEGMENT SYSSMSG;  
CREATE DBNAME ALLDBS.DEFdict  
SEGMENT DEFdict  
SEGMENT SYSSMSG;  
CREATE DBNAME ALLDBS.DEFdb  
SEGMENT USER-SEGMENT1  
SEGMENT USER-SEGMENT2;  
.  
.  
.
```

## 2.4.6 Secondary dictionaries

**Sharing dictionary areas:** If your system has multiple dictionaries defined in 10.2, you may or may not need to define additional DBNAMEs depending on whether those dictionaries share areas:

- If every dictionary has its own DDLDML and DDLDCLOD area and shares the system message area (the SYMSG segment in 12.0), then no additional DBNAME entries are required.
- If two or more dictionaries share the same DDLDCLOD area, then you must place the load area in its own segment and define a DBNAME for each dictionary which shares the load area. Include in the DBNAME definition both the segment that contains the DDLDML area and the segment that contains the shared load area.
- If two or more dictionaries share a DDLCMSG area other than the system message area, then use the approach outlined above to share the message area.

**Different page groups:** If your dictionaries have different page groups (or db-key radices), they cannot share areas. This also applies to the system message area (in segment SYMSG), which can be included only in dictionaries having the same page group.

## 2.4.7 Conflicting area names

**Used 10.2 DMCL ALIAS parameter:** If you have user database areas with conflicting names requiring the use of the ALIAS parameter in your 10.2 DMCL definition, you must define separate DBNAMEs in your 12.0 database name table for each set of conflicting area names. You should not include segments containing conflicting area names in your default DBNAME definition.

**Used DBNAMEs in 10.2:** If you used DBNAMEs in 10.2, then each 10.2 DBNAME will correspond to a 12.0 DBNAME. If, instead of using DBNAMEs in 10.2, your applications dynamically changed the name of the subschema to which they bound the run unit, then you must use subschema mapping rules on the DBTABLE statement to achieve the same result in 12.0.

For example, if an application program binds a run unit using either subschema EMPE01 or EMPW01, depending on whether it needs to access eastern region employee data or western region, then the 12.0 DBTABLE statement would contain the following mapping rules:

```
SUBSCHEMA EMPE01 MAPS TO EMPE01 USING DBNAME EMPEAST  
SUBSCHEMA EMPW01 MAPS TO EMPW01 USING DBNAME EMPWEST
```

You must define DBNAMEs for EMPEAST and EMPWEST identifying the segment (or segments) containing the eastern or western region data.

**Subschema naming standards:** If you use naming standards for your subschemas, you can generalize the above mapping rules as:

```
SUBSCHEMA EMPE???? MAPS TO EMPE???? USING DBNAME EMPEAST  
SUBSCHEMA EMPW???? MAPS TO EMPW???? USING DBNAME EMPWEST
```

You might also wish to eliminate separate subschemas for the western region. If the schema definitions for both east and west are identical (except for pages ranges in 10.2), then you can eliminate the use of separate subschemas by changing the second mapping rule to:

```
SUBSCHEMA EMPW???? MAPS TO EMPE???? USING DBNAME EMPWEST
```

## 2.4.8 Non-dictionary DBNAMEs

For each DBNAME representing a user database in your 10.2 system, you should define a similarly named DBNAME in your 12.0 database name table.

**Review 10.2 subschema mapping rules:** In most cases, you will no longer require the 10.2 subschema mapping rules within the DBNAME definition. In 10.2, these rules caused the name of the subschema known to the program to be changed to one with the appropriate page ranges. Since 12.0 subschemas don't contain page ranges, one subschema can be used to access many physical implementations of the same logical database. Changing the name of the subschema is useful only if the subschemas are derived from different schema definitions.

**DBNAME specified externally in 10.2:** In your DBNAME definition, you must identify the segments containing the data to be accessed by applications binding to the DBNAME. If all applications specify the DBNAME on the BIND statement, then only segments accessed by those applications need to be included in the DBNAME. If, on the other hand, the DBNAME is specified externally in 10.2 (by using DCUF commands or SYSCONTROL parameters, etc.) then you may need to include additional segments within your DBNAME definition or use subschema mapping rules to ensure that run units bind successfully.

**Example:** To illustrate this point, assume again that we have eastern and western region employee data, but in this case, the selection at run time is made by issuing a DCUF SET DBNAME command before executing the online application. If the application accesses only employee data, then the only segment that needs to be included in each DBNAME is the one containing the corresponding employee data. (In fact, if the segments are named the same as the 10.2 DBNAMEs, then you don't even need to define DBNAMEs.)

However, if the application also accesses corporate-wide insurance information (stored in its own segment) then:

1. If the insurance information is accessed in the same run unit as the employee information, include the segment containing the insurance information in both DBNAMEs
2. If the insurance information is accessed in a separate run unit, either:
  - Include the insurance segment in both DBNAMEs or

- Use subschema mappings to redirect the insurance run unit to a different DBNAME

**Redirecting run unit to different DBNAME:** To redirect the run unit to a different DBNAME using subschema mappings, specify the following in the DBNAME definitions for EMPEAST and EMPWEST:

```
SUBSCHEMA EMP????? MAPS TO EMP?????  
SUBSCHEMA ??????? USES DBTABLE MAPPING
```

These parameters have the effect of treating run units binding to subschemas other than those beginning with EMP, as if no DBNAME were specified, hence causing the DBNAME to be selected based on the subschema mapping rules associated with the DBTABLE statement. If you defined your DBTABLE as described earlier under "Basic DBTABLE definition", then unless a more specific mapping rule applies, all run units will use the default DBNAME which should include the insurance segment.

#### 2.4.9 Mixed page groups

If you have segments in different page groups (or having different db-key radices), then there are some additional considerations.

In 12.0, just as in 10.2, you cannot access areas in different page groups within a single run unit. Any attempt to do this will result in a bind failure. The best way to avoid this type of error is to ensure that all segments associated with a DBNAME are in the same page group.

**Note:** CA-IDMS 14.1 lets you specify MIXED PAGE GROUP BINDS ALLOWED for a DBNAME. When this option is used, a run unit can bind to a DBNAME that contains mixed page groups. See the *CA-IDMS Features Guide — Release 14.1* and *CA-IDMS Database Administration* for more details.

CA-IDMS does not prevent mixing segments with different page groups in a DBNAME because there are conditions under which it is desirable to do so. In fact, it is possible that the basic DBTABLE defined above might have segments with different page groups within the default DBNAME. Provided that no one subschema references areas from segments in different page groups (unlikely given the initial assumptions), then this presents no problem.

You can detect potential problems by using the IDMSLOOK utility (or the DC/UCF LOOK task). The DMCL option will warn you if you have mixed page groups within any of your DBNAMEs. The BIND SUBSCHEMA option will notify you of any errors encountered in binding a specific subschema to a specific DBNAME.



# **Chapter 3. Security Migration**

---

3.1 About this chapter . . . . .	3-3
3.2 About security definition migration . . . . .	3-4
3.3 Convert security definitions . . . . .	3-6
3.4 Create execution JCL for the RHDCSMIG program . . . . .	3-7
3.4.1 OS/390 JCL . . . . .	3-7
3.4.2 VSE/ESA JCL . . . . .	3-9
3.4.3 VM/ESA commands . . . . .	3-10
3.4.4 BS2000/OSD JCL . . . . .	3-11
3.5 Output from RHDCSMIG . . . . .	3-13
3.6 Review and modify output from RHDCSMIG . . . . .	3-20
3.7 Populate 12.0 system with security definitions . . . . .	3-22



## 3.1 About this chapter

**The CA-IDMS central security facility:** The CA-IDMS central security facility enables you to control access to the resources in your CA-IDMS environment. You can use this facility to secure all CA-IDMS run-time components, when running under the central version or when running in local mode. You can also use the central security facility to provide definition-time protection for system generation, physical database, and SQL-defined entities.

You will continue to use dictionary security to protect the definitions of schemas, subschemas, records, and other dictionary entities.

The CA-IDMS central security facility controls access to resources using CA-IDMS facilities or an external security package such as CA-ACF2 and CA-Top Secret.

**Note:** You should be familiar with CA-IDMS central security concepts before migrating security definitions. See *CA-IDMS Security Administration* for information on the central security facility.

**Security definition migration:** If you choose to use the IDMS security facility to protect your CA-IDMS resources, a conversion utility is provided that creates initial 12.0 security definitions from your 10.2 system definitions.

This chapter describes how to use the security definition migration utility to convert Release 10.2 security definitions to Release 12.0 security statements.

## 3.2 About security definition migration

**What is the security definition migration utility:** The security definition migration utility is a program that converts user, task, and program security definitions from a 10.2 dictionary to preliminary 12.0 security definitions and authorizations. After executing the security migration utility, review the security statements and modify them, as necessary, to reflect your security requirements.

**What the security migration utility does:** The security definition migration utility reads the records containing security definitions for users, tasks, and programs for a specified DC/UCF system from a 10.2 dictionary. It then creates the appropriate security statements to define resources and, as appropriate, grant the authority to use resources.

Specifically, the security migration utility reads security definitions from these records in a 10.2 DDLDML area:

- ACCESS-045
- SYS-041
- PROGLST-049
- TASKLST-023
- USER-047

The security definition migration utility is run in retrieval mode so that your 10.2 dictionary is not updated.

You should execute the utility against your 10.2 dictionary *before* migrating it to a 12.0 format.

**What the security migration utility produces:** For a specified DC/UCF system, the security definition migration utility produces the following statements:

- A CREATE RESOURCE SYSTEM SYST<sub>nnnn</sub> statement for the DC/UCF system processed.
- A CREATE USER statement for each user ID defined with SIGNON authority to the DC/UCF system.
- A CREATE SYSTEM PROFILE statement for each such user ID having INSTALLATION CODE or PRIORITY clauses assigned.
- A GRANT SIGNON ON SYSTEM SYST<sub>nnnn</sub> statement for each user having SIGNON authority to the DC/UCF system. A PROFILE clause is only included if the user ID has INSTALLATION CODE or PRIORITY clauses associated with it.
- A CREATE RESOURCE CATEGORY CAT\_<sub>nnn</sub> statement for only those security classes assigned to a task or program.
- A GRANT EXECUTE ON CATEGORY CAT\_<sub>nnn</sub> TO *user-id* statement for each user ID associated with the specified category (security class).

- A CREATE RESOURCE ACTIVITY DEFAULT.ACT\_*nnn* NUMBER *nnn* statement for each CA-ADS security class (1 through 255).
- A GRANT EXECUTE ON ACTIVITY ... TO *user-id* statement for each user ID associated with the specified activity (security class).

Sample output and a more detailed description of the output is provided later in this chapter.

**How to execute the security definition migration utility:** The security definition migration utility executes as program RHDCSMIG and uses the RHDCMIGA subschema.

Your 12.0 DMCL definition must contain the 10.2 file and area description for the 10.2 DDLDML area that will be processed by the security definition migration utility. The security migration utility only runs in a Release 12.0 environment.

You must execute the RHDCSMIG program against a 10.2 dictionary. You cannot run RHDCSMIG against a DDLDML area that has been migrated to a Release 12.0 format.

**Conversion tools and environment:** To use the security migration utility and generate your 12.0 central security definitions, you need:

This component	For these tasks
12.0 DMCL module	Run-time environment needed to execute the security migration utility (RHDCSMIG).
RHDCSMIG program and RHDCMIGA subschema	To convert security definitions.
10.2 DDLDML area containing security definitions to be converted	The RHDCSMIG program reads security definitions from a DDLDML area.
CA-IDMS Command Facility	To populate the 12.0 user catalog and DDLDML area with modified 12.0 security statements.

**Related CA documentation:** You may need to refer to these documents during the process of migrating 10.2 security definitions:

- *CA-IDMS Security Administration*
- *CA-IDMS Database Administration*

### 3.3 Convert security definitions

To convert security definitions using the security definition migration utility, take these steps:

1. Identify the 10.2 DDLDML area that contains the security definitions to be converted.
2. Verify that this DDLDML area is defined in the 12.0 DMCL you'll use to execute the security definition migration utility.
3. Create execution JCL for the RHDCSMIG program.
4. Execute RHDCSMIG program.
5. Review security statements created by the RHDCSMIG program.
6. Modify security statements.
7. Submit security statements to the CA-IDMS Command Facility to populate both the user catalog and the DDLDML area.

Some of these steps are discussed in more detail below.

## 3.4 Create execution JCL for the RHDCSMIG program

**Runs in local mode or under the central version:** The RHDCSMIG program can run either in local mode or under the central version. It is recommended that you run the program in local mode.

**Input to RHDCSMIG:** Input to RHDCSMIG is the *dc/ucf-version-number* of the DC/UCF system whose definition will be read by the RHDCSMIG program.

You specify the *dc/ucf-version-number* as a SYSIPT parameter as follows:

- DCSYSTEM=*nnnn*. There can be no intervening spaces between DCSYSTEM and *nnnn*.
- The DCSYSTEM keyword can begin in any column and must be the first record in the SYSIPT file.
- *Nnnn* must be an integer in the range 1 through 9999. Leading zeroes are not required.

The RHDCSMIG program produces a SYSPCH output file containing the 12.0 security statements.

An internal sort is performed during RHDCSMIG processing, so sort work files must be provided in the JCL.

The execution JCL required to run the RHDCSMIG is provided below.

### 3.4.1 OS/390 JCL

To run the RHDCSMIG program in an OS/390 environment in local mode, code the following execution JCL.

**Local mode:**

```

//RUNSMIG EXEC PGM=RHDCSMIG,REGION=1024K
//STEPLIB  DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib,DISP=SHR
//sysjrn1  DD DUMMY
//SORTWK01 DD UNIT=disk,SPACE=(CYL,(1,1))
//SORTWK02 DD UNIT=disk,SPACE=(CYL,(1,1))
//SORTWK03 DD UNIT=disk,SPACE=(CYL,(1,1))
//SORTWK04 DD UNIT=disk,SPACE=(CYL,(1,1))
//SORTMSG  DD SYSOUT=A
//dictdb   DD DSN=cdms.dictdb,DISP=SHR
//dcmmsg   DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR
//SYSLST   DD SYSOUT=A
//SYSPCH   DD DSN=your.security.syntax,DISP=(NEW,CATLG,DELETE),
//           DCB=(LRECL=80,BLKSIZE=9040,RECFM=FB,DSORG=PS),
//           UNIT=3380,SPACE=(TRK,(50,10),RLSE),VOL=SER=vvvvv
//SYSOUT   DD SYSOUT=A
//SYSIDMS  DD *
DMCL=dmcl-name
DBNAME=dbname or segment-name

Add other SYSIDMS parameters, as appropriate.

/*
//SYSIPT   DD  *
DCSYSTEM=nnnn
/*
*/

```

<u>idms.dba.loadlib</u>	Data set name of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Data set name of the load library containing the CA-IDMS executable Release 12.0 modules
<u>dictdb</u>	DDname of the 10.2 DDLDML area containing the security definitions to be converted. This is the ddname specified in your Release 12.0 DMCL.
<u>cdms.dictdb</u>	Dataset name of the 10.2 DDLDML area containing the security definitions to be converted
<u>dcmmsg</u>	DDname of the system message area (DDLDMSG)
<u>idms.sysmsg.ddldcmsg</u>	Dataset name of the system message area (DDLDMSG)
<u>your.security.syntax</u>	Dataset name where the resultant 12.0 security syntax will be placed
<u>vvvvvv</u>	Label of disk volume on which the output dataset will reside
<u>dmcl-name</u>	Name of the Release 12.0 DMCL module
<u>dbname</u> or <u>segment-name</u>	Database name or segment name containing the DDLDML area to be migrated

---

<u>nnnn</u>	An integer from 1 through 9999 that identifies the DC/UCF system whose definition will be read by the RHDCSMIG program. This is the <i>dc/ucf-version-number</i> assigned to the DC/UCF system when it was defined.
-------------	---

---

**Central version:** To run the RHDCSMIG program under the central version, modify the local mode JCL as follows:

- Add a SYSCTL file
- Remove file definition for the DDLDML area

### 3.4.2 VSE/ESA JCL

To run the RHDCSMIG program in a VSE/ESA environment in local mode, code the following execution JCL.

```
// EXEC PROC=IDMSLBLs
// DBLBL SORTWK1'<u>sort.file.SORTWK01</u>'
// EXTENT SYSnnn,nnnnn,,ssss,100
// ASSGN SYSnnn,DISK,VOL=vvvvv,SHR
// DBLBL SORTWK2'<u>sort.file.SORTWK02</u>'
// EXTENT SYSnnn,nnnnn,,ssss,100
// ASSGN SYS120,DISK,VOL=CULLD2,SHR
// DBLBL SORTWK3'<u>sort.file.SORTWK03</u>'
// EXTENT SYSnnn,nnnnn,,ssss,100
// ASSGN SYS120,DISK,VOL=CULLD2,SHR
// DBLBL SORTWK4'<u>sort.file.SORTWK04</u>'
// EXTENT SYSnnn,nnnnn,,ssss,100
// ASSGN SYS120,DISK,VOL=CULLD2,SHR
// DBLBL IJSYSPH,'<u>syspch.work.area</u>',0,SD
// EXTENT SYSPCH,nnnnn,,ssss,100
// ASSGN SYSnnn,DISK,VOL=vvvvv,SHR
// DBLBL <u>idmslib,'idms.library.</u>',1999/365
// EXTENT ,nnnnn,,ssss,1500
// LIBDEF *,SEARCH=<u>idmslib.sublib</u>
// EXEC RHDCSMIG
DMCL=dmc1-name
DBNAME=<u>dbname or segmen-name</u>
/*
DCSYSTEM=<u>nnnn</u>
/*
CLOSE SYSPCH,00D
/*
```

---

<u>sortwknn</u>	Name of the sort work file
<u>idmslib.sublib</u>	Name of the sublibrary within the library containing CA-IDMS modules
<u>idmslib</u>	Filename of the file containing CA-IDMS modules
<u>idms.library</u>	File-ID associated with the file containing CA-IDMS modules

---

<u>SYSnnn</u>	Logical unit of the volume for which the extent is effective
<u>nnnnnn</u>	Volume serial identifier of appropriate disk volume
<u>ssss</u>	Starting track (CKD) or block (FBA) of disk extent
<u>dmcl</u>	Name of the DMCL module
<u>dbname or segment-name</u>	Name of the database or segment containing the DDLDML area to be processed by RHDCSMIG

**Central version:** To run the RHDCSMIG program under the central version, modify the local mode JCL as follows:

- Add a SYSCTL file
- Remove file definition for the DDLDML area

### 3.4.3 VM/ESA commands

```

FILEDEF SYSLST PRINTER
FILEDEF SORTMSG PRINTER
FI dictdb DISK vaddr
FI dcmsg DISK vaddr
FI SYSPCH DISK security syntax a
FILEDEF SYSIDMS DISK sysidms input a
FILEDEF SYSIPT DISK rhdcsmig input a
GLOBAL LOADLIB dbalib idmslib
OSRUN RHDCSMIG

```

<u>dictdb</u>	DDname of the 10.2 DDLDML area containing the security definitions to be converted. This is the ddname specified in your Release 12.0 DMCL.
<u>dcmsg</u>	DDname of the system message (DDLDCMSG) area
<u>vaddr</u>	Virtual address of the mini disk on which the file resides
<u>security syntax a</u>	File identifier of the file where the resultant 12.0 security syntax will be placed
<u>sysidms input a</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ <i>dmcl-name</i> — name of Release 12.0 DMCL</li> <li>■ <i>dbname or segment-name</i> — database name or segment name identifying the DDLDML area to be migrated</li> </ul>

---

<u>rhdcsmig input a</u>	File identifier of the file containing the RHDCSMIG input parameter:
	DCSYSTEM=nnnn — An integer from 1 through 9999 that identifies the DC/UCF system whose definition will be read by the RHDCSMIG program. This is the <i>dc/ucf-version-number</i> assigned to the DC/UCF system when it was defined.
<u>dbalib</u>	Filename of the load library containing the DMCL and database name table load modules

---

<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules
----------------	---

---

**Central version:** To run the RHDCSMIG program under the central version, modify the local mode JCL as follows:

- Add a SYSCTL file
- Remove file definition for the DDLDML area

#### 3.4.4 BS2000/OSD JCL

**Local mode:** To run the RHDCSMIG program in a BS2000/OSD environment in local mode, code the following execution JCL.

```
/ FILE      LINK=CDMSLIB,idms.dbalib
/ FILE      LINK=CDMSLIB1,idms.loadlib
/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSPAM,lodwork,SPACE=(primlodr,seclodr)
/ FILE      LINK=sysjrn1,*DUMMY
/ FILE      LINK=dictdb,SHARUPD=YES,idms.dictdb
/ FILE      LINK=dcmsg,SHARUPD=YES,idms.sysmsg.ddldcmmsg
/ FILE      LINK=SYSIDMS,*DUMMY
/ SYSPFILE  SYSOPT=your.security.syntax
/ EXEC      (RHDCSMIG,idms.loadlib)
DMCL=dmc1-name
DBNAME=dbname or segment name
END-SYSIDMS
DCSYSTEM=nnnn
```

---

<u>idms.dba.loadlib</u>	Filename of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing the CA-IDMS executable Release 12.0 modules

---

### 3.4 Create execution JCL for the RHDCSMIG program

---

<u>dictdb</u>	Linkname of the 10.2 DDLDML area containing the security definitions to be converted. This is the ddname specified in your Release 12.0 DMCL.
<u>idms.dictdb</u>	Filename of the 10.2 DDLDML area containing the security definitions to be converted
<u>dcmsg</u>	Linkname of the system message area (DDLDCMSG)
<u>idms.sysmsg.ddldcmsg</u>	Filename of the system message area (DDLDCMSG)
<u>dmcl-name</u>	Name of the Release 12.0 DMCL module
<u>dbname or segment-name</u>	Database name or segment name containing the DDLDML area to be migrated
<u>DCSYSTEM=nnnn</u>	An integer from 1 through 9999 that identifies the DC/UCF system whose definition will be read by the RHDCSMIG program. This is the <i>dc/ucf-version-number</i> assigned to the DC/UCF system when it was defined.

**Central version:** To run the RHDCSMIG program under the central version, modify the local mode JCL as follows:

- Add a SYSCTL file
- Remove file definition for the DDLDML area

## 3.5 Output from RHDCSMIG

A description of the output from the security definition migration utility is provided below. You should review the output from your execution of the utility and modify it to reflect your security definition requirements. After the sample output is described, a checklist of items to review is provided.

**CREATE RESOURCE SYSTEM statement:** A CREATE RESOURCE SYSTEM SYST $n$ nnn statement is created for the DC/UCF system identified on the DCSYSTEM SYSIPT control card.

*Sample output*

```
CREATE RESOURCE SYSTEM SYST0045;
```

**CREATE USER statement:** A CREATE USER statement is created for each user ID defined with SIGNON authority to the specified DC/UCF system. If the following information was part of the 10.2 USER definition, it will be included in the 12.0 user definition.

12.0 statement component	Created from 10.2 statement
DESCRIPTION 'description'	DESCRIPTION IS clause of the IDD USER statement
NAME <i>user-name</i>	ALIAS IS clause of system generation USER statement or FULL NAME IS clause of IDD USER statement.
ENCRYPTED PASSWORD X'nnnnnnnnnnnnnnnn'	PASSWORD IS from either system generation or IDD USER statement.  The contents of PASSWORD IS is migrated in its encrypted format (X'16-hex-digits'). Once the password is migrated, it can continue to be used as is or changed. The password is displayed in its encrypted format by RHDCSMIG.

**CREATE SYSTEM PROFILE:** A CREATE SYSTEM PROFILE *user-id\_nnnn* statement is created for each user ID with SIGNON authority to the specified DC/UCF system. *Nnnn* is the DC/UCF system number.

The following information is included in the CREATE SYSTEM PROFILE statement if it was defined in the 10.2 USER definition.

12.0 statement component	Created from 10.2 statement
INSTCODE	INSTALLATION CODE from system generation or IDD USER statement.
PRIORITY	PRIORITY IS from system generation or IDD USER statement.

*Sample CREATE USER and SYSTEM PROFILE output*

```
CREATE USER "AAD260          "
    ENCRYPTED PASSWORD X'FA808D8EDB038200'
;
CREATE SYSTEM PROFILE "AAD260_0045      " ATTRIBUTES
    INSTCODE='AD260           ' OVERRIDE NO
;
CREATE USER "AAE120          "
    ENCRYPTED PASSWORD X'C0F2F9C551A73D12'
;
CREATE SYSTEM PROFILE "AAE120_0045      " ATTRIBUTES
    INSTCODE='AE120           ' OVERRIDE NO
;
CREATE USER "CULL DBA        "
    ENCRYPTED PASSWORD X'E57C572D47B0D600'
;
CREATE SYSTEM PROFILE "CULL DBA_0045      " ATTRIBUTES
    PRIORITY=050 OVERRIDE NO
;
```

**GRANT SIGNON ON SYSTEM ... TO USER:** A GRANT SIGNON ON SYSTEM SYST<sub>nnnn</sub> statement is created for each user ID with SIGNON authority to the specified DC/UCF system. This gives the user ID authority to sign on to the specified DC/UCF system.

The PROFILE "user-id\_nnnn" clause is included if INSTALLATION CODE or PRIORITY clauses were part of the 10.2 user definition.

*Sample output*

```
GRANT SIGNON ON SYSTEM SYST0045
    PROFILE "AAD260_0045      "
    TO "AAD260           ";
GRANT SIGNON ON SYSTEM SYST0045
    PROFILE "AAE120_0045      "
    TO "AAE120           ";
GRANT SIGNON ON SYSTEM SYST0045
    PROFILE "CULL DBA_0045      "
    TO "CULL DBA           ";
```

**CREATE RESOURCE CATEGORY:** A CREATE RESOURCE CATEGORY CAT<sub>nnn</sub> statement is created for each security class used by a task or program in the 10.2 dictionary. ADD TASK, ADD PROGRAM, and ADD LOAD MODULE statements are included for each CREATE RESOURCE CATEGORY statement.

If a security class is not used by a task or program, a CREATE RESOURCE CATEGORY statement is not created for the security class. However, if a security class is not used by any task or program and is associated with a user ID, a GRANT EXECUTE ON CATEGORY CAT\_*nnn* TO "user-id" statement is still created by the security definition migration utility.

For more information, see "GRANT EXECUTE ON CATEGORY" below.

<b>12.0 statement component</b>	<b>Created from 10.2 statement</b>
CREATE RESOURCE CATEGORY CAT_ <i>nnn</i>	SECURITY CLASS clause in TASK or PROGRAM statement.
ADD TASK	System generation and IDD ADD TASK statement and SECURITY CLASS clause.
ADD PROGRAM	System generation PROGRAM statement for programs defined with either the DYNAMIC or LOADLIB parameters.  RHDCSMIG creates a prefix to the program name as follows: <ul style="list-style-type: none"><li>■ Version 1 is prefixed with CDMSLIB.</li><li>■ Other version numbers for the same program are prefixed with V<i>nnnn</i> where <i>nnnn</i> is the 10.2 version number for the program.</li></ul>
ADD LOAD MODULE	System generation PROGRAM statement for programs defined as either DYNAMIC with a type other than PROGRAM or with an associated DICTNAME.  The RHDCSMIG program prefixes the load module name with V <i>nnnn</i> where <i>nnnn</i> is the 10.2 version number for the program.

**GRANT EXECUTE ON CATEGORY:** A GRANT EXECUTE ON CATEGORY CAT\_*nnn* TO "user-id" statement is created for each user ID with assigned security classes. Note that this statement is created even if the security class is not assigned to a TASK or PROGRAM definition (and therefore will not have a corresponding CREATE RESOURCE CATEGORY statement).

**Note:** An error message is returned for those GRANT EXECUTE ON CATEGORY statements (security classes) for which there is no preceding CREATE RESOURCE CATEGORY statement (not assigned to a task or program statement) and the dictionary is not populated with the GRANT EXECUTE statements. You can ignore the error messages or delete the GRANT EXECUTE statements.

**Important:** If signon definitions are maintained in a dictionary separate from the dictionary containing system generation definitions, you must run the RHDCSMIG program separately on both dictionaries and merge the output. In this case, do not delete any statements until after you merge and review the output from RHDCSMIG. For more information, see 3.6, “Review and modify output from RHDCSMIG” on page 3-20.

12.0 statement component	Created from 10.2 statement
GRANT EXECUTE ON CATEGORY CAT_ <i>nnn</i> TO " <i>user-id</i> "	SECURITY CLASS clause of system generation USER statement or  INCLUDE ACCESS of IDD ADD USER statement.
GRANT EXECUTE ON CATEGORY CAT_* TO " <i>user-id</i> "	A wild-card character (*) is appended to the category name to assign all categories to a user ID if all security classes were previously assigned to the user.

*Sample output*

```
CREATE RESOURCE CATEGORY CAT_001
  ADD TASK ADS
  ADD TASK ASF
  ADD TASK ASFNT
  ADD TASK A303EIS
  ADD TASK CLIST
  ADD TASK DCUF
  ADD TASK ICMS
  ADD TASK ICMSCOMM
  ADD TASK IDB
  ADD TASK IDBCOMM
  ADD TASK IDD
  ADD TASK IDDM
  ADD TASK IDDMT
  ADD TASK IDDT
  ADD TASK OLQ
  ADD TASK OLQNT
  ADD TASK OLQT
  ADD TASK SEND
  ADD TASK SHOWMAP
  ADD TASK TCF
;
GRANT EXECUTE ON CATEGORY CAT_001 TO "AAD260"      ";
GRANT EXECUTE ON CATEGORY CAT_001 TO "AAE120"      ";
CREATE RESOURCE CATEGORY CAT_002
  ADD TASK DCMT
  ADD TASK DCPROFIL
;
GRANT EXECUTE ON CATEGORY CAT_002 TO "AAD260"      ";
GRANT EXECUTE ON CATEGORY CAT_002 TO "AAE120"      ";
.
.
.

CREATE RESOURCE CATEGORY CAT_099
  ADD PROGRAM CDMSLIB.DBUGMAIN
  ADD PROGRAM V0002.MYMAP
  ADD LOAD MODULE V0002.MYMAP
  ADD TASK ADSA
  ADD TASK ADSAT
  ADD TASK ADSD
  ADD TASK ADSG
  ADD TASK ADSGT
  ADD TASK ADSOTATU
  ADD TASK DEBUG
  ADD TASK MRIDEFQ
  ADD TASK MRIMSGQ
  ADD TASK MRISTAQ
  ADD TASK OLM
  ADD TASK OLMT
;
```

**CREATE RESOURCE ACTIVITY:** A CREATE RESOURCE ACTIVITY  
DEFAULT.ACT\_*nnn* NUMBER *nnn* is created for all security classes from 1 through  
255. *Nnn* is the security class from the 10.2 dictionary.

**GRANT EXECUTE ON ACTIVITY:** The GRANT EXECUTE ON ACTIVITY DEFAULT.ACT\_*nnn* TO "*user-id*" statement gives the specified user ID EXECUTE authority on the specified CA-ADS, DCMT, or Online debugger (DEBUG) activity. *Nnn* is the security class from the 10.2 dictionary. A GRANT EXECUTE ON ACTIVITY... TO "*user-id*" statement is created for each user ID assigned the specified security class.

12.0 statement component	Created from 10.2 statement
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_ <i>nnn</i> to " <i>user-id</i> "	SECURITY CLASS clause of system generation ADD USER statement or
You may wish to refine the access to the activity to a particular CA-ADS application, the DCMT command facility, or the Online debugger. If so, define additional activities in which you replace DEFAULT with the name of the application: <ul style="list-style-type: none"><li>■ CA-ADS</li><li>■ DCMT</li><li>■ DEBUG</li></ul>	INCLUDE ACCESS of IDD ADD USER statement

*Sample output*

```
CREATE RESOURCE ACTIVITY DEFAULT.ACT_001 NUMBER 001;
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_001 TO "AAD260"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_001 TO "AAE120"      ";
CREATE RESOURCE ACTIVITY DEFAULT.ACT_002 NUMBER 002;
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_002 TO "AAD260"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_002 TO "AAE120"      ";
CREATE RESOURCE ACTIVITY DEFAULT.ACT_003 NUMBER 003;
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_003 TO "AAD260"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_003 TO "AAE120"      ";
CREATE RESOURCE ACTIVITY DEFAULT.ACT_004 NUMBER 004;
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_004 TO "AAD260"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_004 TO "AAE120"      ";
CREATE RESOURCE ACTIVITY DEFAULT.ACT_005 NUMBER 005;
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_005 TO "AAD260"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_005 TO "AAE120"      ";
GRANT EXECUTE ON ACTIVITY DEFAULT.ACT_*    TO "CULL DBA"      ";

...
CREATE RESOURCE ACTIVITY DEFAULT.ACT_243 NUMBER 243;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_244 NUMBER 244;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_245 NUMBER 245;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_246 NUMBER 246;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_247 NUMBER 247;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_248 NUMBER 248;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_249 NUMBER 249;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_250 NUMBER 250;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_251 NUMBER 251;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_252 NUMBER 252;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_253 NUMBER 253;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_254 NUMBER 254;
CREATE RESOURCE ACTIVITY DEFAULT.ACT_255 NUMBER 255;
```

## 3.6 Review and modify output from RHDCSMIG

You should review the 12.0 security statements created by RHDCSMIG and modify them to reflect your security requirements. Here are some considerations:

1. Under 10.2 security, all versions of a program had the same security class. Under 12.0 security, each version of a program is controlled separately. You may need to add additional PROGRAM or LOAD MODULE clauses to your CATEGORY definitions in order to achieve identical results. You should also consider using the wild-card capability to reduce the number of definitions and to provide for load modules which are defined dynamically.
2. Under 10.2 security, all tasks and programs for which no security class was assigned were considered "unsecured," indicating that anyone could execute them. To achieve similar results under 12.0, you can issue the following statements:

```
CREATE RESOURCE CATEGORY category-name
  ADD PROGRAM *
  ADD TASK *
  ADD LOAD MODULE *;
```

```
GRANT EXECUTE ON CATEGORY category-name to PUBLIC;
```

3. If you maintain user signon definitions in a dictionary separate from the dictionary containing system generation definitions, you need to run the RHDCSMIG program once against each dictionary and then merge the output files.

Review the combined output and make the necessary changes to the combined file.

For example, you will have duplicate CREATE SYSTEM RESOURCE statements and CREATE RESOURCE ACTIVITY statements.

4. Review the generated system name. It should be the same as the system name identified on the new system generation SYSTEM statement SYSTEM ID IS parameter. The dictionary migration utility will generate a value for the SYSTEM ID IS parameter as follows:

- If a 10.2 system generation DDS statement LOCAL NODE IS clause is defined, the value specified on this clause will become the 12.0 system name or
- If a 10.2 DDS statement is not specified, the system name defaults to SYST $nnnn$  where  $nnnn$  is the value specified as *dc/ucf-version* on the system generation ADD SYSTEM statement

The system name on the CREATE RESOURCE SYSTEM statement and the SYSTEM ID IS parameter of the system generation SYSTEM statement must be the same value.

5. Review system profile names and category names created by RHDCSMIG. If necessary, change the names generated to reflect your naming standards.

If multiple users have the same priority and installation code values, consider having those users share a single profile.

6. To avoid errors, identify any GRANT EXECUTE CATEGORY statements for which there is no corresponding CREATE RESOURCE CATEGORY statement and delete them.
7. Review CREATE RESOURCE ACTIVITY and corresponding GRANT EXECUTE ON ACTIVITY statements. If you want to refine the access to the activity to a particular CA-ADS application, DCMT command, or the online debugger, define additional sets of activities in which you replace the prefix DEFAULT with the name of the application (CA-ADS, DCMT, or DBUG).
8. If multiple DC/UCF systems are defined in the same dictionary, they will all have the same security characteristics. A task or program is associated with the same category in all systems and users have the right to execute the same categories in all systems. If this is not desired, you must define the systems in separate dictionaries.

## 3.7 Populate 12.0 system with security definitions

**Submit statements to the command facility:** To populate the DDLDML area and the user catalog with your security definitions, submit the statements to the CA-IDMS Command Facility.

All security statements except the CREATE USER statement are stored in the DDLDML area. CREATE USER statements are stored in the user catalog.

**Connect to the system dictionary:** For security definitions to be effective, you must access the SYSTEM dictionary when submitting statements. Be sure you are connected to the SYSTEM dictionary.

User definitions are automatically stored in the user catalog through a connect that is hard-coded in the CA-IDMS software.

►►For information on using the CA-IDMS Command Facility, see *CA-IDMS Command Facility*.

**Review listing from the command facility:** Review the listing produced by the Command Facility and, if necessary, correct any errors.

**Security Display Facility:** To generate a report from the user catalog on the security definition for a DC/UCF system, you can use the Security Display Facility to generate an online or batch report.

►►For information on using the Security Display Facility, see *CA-IDMS Security Administration*.

# Chapter 4. Dictionary Migration and Setup

---

4.1 About this chapter . . . . .	4-3
4.2 About the 12.0 dictionary environment . . . . .	4-4
4.3 About dictionary migration . . . . .	4-5
4.4 Planning to migrate dictionaries . . . . .	4-7
4.5 Migrate dictionaries . . . . .	4-8
4.5.1 The dictionary migration process . . . . .	4-8
4.5.2 Running RHDCMIG1 . . . . .	4-8
4.5.2.1 OS/390 execution JCL . . . . .	4-8
4.5.2.2 VSE/ESA execution JCL . . . . .	4-9
4.5.2.3 VM/ESA commands . . . . .	4-10
4.5.2.4 BS2000/OSD execution JCL . . . . .	4-11
4.5.3 Verify the results of RHDCMIG1 . . . . .	4-12
4.5.4 Running RHDCMIG2 . . . . .	4-12
4.5.4.1 OS/390 JCL . . . . .	4-13
4.5.4.2 VSE/ESA JCL . . . . .	4-14
4.5.4.3 VM/ESA commands . . . . .	4-14
4.5.4.4 BS2000/OSD JCL . . . . .	4-15
4.5.5 Verify the results of RHDCMIG2 . . . . .	4-16
4.6 Run Release 12.0 IDMSDIRL utility . . . . .	4-17
4.7 Update the Task Application Table . . . . .	4-18
4.8 Set up 12.0 dictionary environment . . . . .	4-19
4.8.1 Setting up a system dictionary . . . . .	4-19
4.8.2 Setting up application dictionaries . . . . .	4-20
4.8.3 Updating the load (DDLDCLOD) area . . . . .	4-20
4.9 Modify schemas and regenerate subschemas . . . . .	4-21
4.9.1 Using IDMSSCON to convert 10.0 subschema load modules . . . . .	4-22
4.9.2 Running IDMSSCON . . . . .	4-22
4.9.2.1 OS/390 execution JCL . . . . .	4-23
4.9.2.2 VSE/ESA execution JCL . . . . .	4-24
4.9.2.3 VM/ESA commands . . . . .	4-24
4.9.2.4 BS2000/OSD execution JCL . . . . .	4-25
4.9.3 Verifying results of IDMSSCON . . . . .	4-26



## 4.1 About this chapter

The structure of the Release 12.0 dictionary environment has changed to accommodate the features of CA-IDMS Release 12.0. This chapter describes the changes to the 12.0 dictionary environment and provides guidelines for migrating 10.0 dictionaries and setting up a 12.0 dictionary environment.

## 4.2 About the 12.0 dictionary environment

**Migrate only the DDLDML area:** The DDLDML area has changed and you must migrate it to the new Release 12.0 format. CA-IDMS provides a dictionary migration utility to migrate 10.0 DDLDML areas to their 12.0 format. For a list of the changes to the dictionary, see *CA-IDMS Dictionary Structure Reference Guide*.

**Unchanged dictionary areas:** These dictionary areas have not changed and you can use them in a 12.0 environment without any *structural* changes:

- DDLCLOD
- DDLCMSG

**Note:** You must populate your message area with CA-IDMS 12.0 messages.

**New dictionary areas:** There are new dictionary areas and they are installed during the CA-IDMS installation process. They are:

- DDLCAT — Contains physical database entities. If you have the SQL Option, it will also contain SQL database entities.
- DDLCATX — Contains indexes associated with DDLCAT entities
- DDLCATLOD — Contains load modules for DMCLs, database name tables and access modules (SQL Option only)

## 4.3 About dictionary migration

**The dictionary migration utility:** CA-IDMS provides a dictionary migration utility that migrates the DDLDML area of Release 10.0 data dictionaries to the new 12.0 format.

**What the dictionary migration utility does:** The dictionary migration utility makes these changes to DDLDML areas:

- Deletes records, elements, and set structures no longer needed in the 12.0 environment
- Restructures record occurrences from a 10.0 format to a 12.0 format
- Adds new record elements and set structures

The dictionary migration utility:

- Migrates only the DDLDML area of Release 10.0 data dictionaries

You cannot migrate dictionaries prior to Release 10.0 with the 12.0 dictionary migration utility.
- Executes in a Release 12.0 environment only
- Executes in local mode only
- Executes as two programs:
  - Program RHDCMIG1 — Deletes and modifies records, elements, and set pointers
  - Program RHDCMIG2 — Deletes and modifies records
- ***Does nothing to CA-IDMS/DB databases. Do not migrate your CA-IDMS/DB databases.***

**When to migrate dictionaries:** Once you have processed your 10.2 DMCL definitions using the DMCL Syntax Generator and produced your 12.0 security definitions using the RHDCSMIG utility program, you can then migrate the DDLDML area of your 10.0 data dictionaries. For details on using the DMCL Syntax Generator, see Chapter 2, “Physical Database Definition.”

**Conversion tools and environment:** You will need the following components to migrate 10.0 DDLDML areas:

This component	For these tasks
RHDCMIG1 and RHDCMIG2 programs and RHDCMIGA and IDMSNWKA (12.0 version) subschemas.	To migrate DDLDML areas.
These modules are installed during the installation process and should exist in the load (core-image) library that contains CA-IDMS executable modules.	
RHDCMIGA is a subschema load module used by RHDCMIG1 that describes a 10.0 dictionary.	
IDMSNWKA is a 12.0 subschema that describes the 12.0 DDLDML area.	
Release 10.0 DDLDML areas.	These are the dictionary areas the migration utility will migrate. You must define these DDLDML areas in the 12.0 DMCL you'll use to run the migration utility.
Release 12.0 DMCL load module, 12.0 database name table (if one is named by the 12.0 DMCL), and the DBA and LOADLIB load libraries.	To run RHDCMIG1 and RHDCMIG2 in a 12.0 environment.

**Related CA documentation:** You may need to refer to these documents during the process of migrating dictionaries:

- *CA-IDMS Database Administration*
- *CA-IDMS Utilities*
- *CA-IDMS Installation and Maintenance Guide - OS/390*

## 4.4 Planning to migrate dictionaries

Before you migrate any dictionaries, make sure you have the correct 10.2 and 12.0 environments set up to properly migrate DDLDML dictionary areas.

Here are some guidelines.

1. If the integrity of any of your DDLDML areas is questionable, you may want to run IDMSDBAN against them to verify that there are no integrity problems. The dictionary migration utility will not run against a DDLDML area with broken chains.
2. Evaluate the space available in each 10.0 DDLDML area you will migrate by reviewing the file utilization report produced by the IDMSDUMP utility.
3. If you need more space, expand the size of the area by either increasing the page size using IDMSXPAG or increasing the page size and/or page range within the area by using the IDMSUNLD and IDMSDBLU utilities.

The dictionary migration process **does not** require that you increase the size of the DDLDML. However, IDMSDIRL will require approximately 5% additional space.

If you increase the size of the DDLDML area, remember to modify your 12.0 DMCL to reflect this change.

4. Backup each DDLDML area you plan to migrate.
5. Make sure the Release 12.0 DMCL you'll be using to execute the dictionary migration utility includes the segments containing the DDLDML areas you will migrate. The name of each DDLDML area must be DDLDML. If you are migrating multiple DDLDML areas, they must be associated with different segments.

## 4.5 Migrate dictionaries

### 4.5.1 The dictionary migration process

Once you've prepared your environment to migrate your dictionaries, perform these steps:

1. Backup the DDLDML area to be migrated if you haven't done so already.
2. Code execution JCL for RHDCMIG1.
3. Run RHDCMIG1 against the 10.0 DDLDML area. RHDCMIG1 runs from the 12.0 load (core-image) library that contains CA-IDMS executable load modules.
4. Verify results by reviewing the transaction summary.
5. If the DDLDML area you are migrating is large or in your estimation took a long time to migrate during phase I of the migration process, you may want to backup the area before running RHDCMIG2.
6. Code execution JCL for RHDCMIG2.
7. Run RHDCMIG2.
8. Verify results by reviewing messages provided by RHDCMIG2.
9. If the DDLDML area you've just migrated contains the IDMSNTWK schema, run the 12.0 IDMSDIRL utility against it once RHDCMIG1 and RHDCMIG2 have successfully executed.

**What's next:** The remainder of this section presents

- Execution JCL to run RHDCMIG1 and RHDCMIG2
- Guidelines for running Release 12.0 IDMSDIRL
- Guidelines for setting up the rest of your dictionary environment

### 4.5.2 Running RHDCMIG1

The execution JCL to run RHDCMIG1 in local mode is provided below.

#### 4.5.2.1 OS/390 execution JCL

To run RHDCMIG1 in local mode, code the following execution JCL.

```

//RHDCMIG1 EXEC PGM=RHDCMIG1,REGION
//STEPLIB   DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib,DISP=SHR
//dictdb    DD DSN=cdms.dictdb,DISP=SHR
//dcmmsg    DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR
//SYSLST    DD SYSOUT=A
//SYSUDUMP  DD SYSOUT=A      OPTIONAL
//SYSIDMS   DD *

Insert SYSIDMS parameters here. Code them on the same or different
lines.

DMCL=dmc1-name
DBNAME=dbname or segment-name

```

<u>idms.dba.loadlib</u>	Data set name of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Data set name of the load library containing the executable Release 12.0 modules
<u>dictdb</u>	DDname of the 10.0 DDLDML area you will migrate. This is the ddname specified in your Release 12.0 DMCL.
<u>cdms.dictdb</u>	Data set name of the 10.0 DDLDML area you will migrate
<u>dcmsg</u>	DDname of the system message (DDLDMSG) area
<u>idms.sysmsg.ddldcmsg</u>	Data set name of the system message (DDLDMSG) area
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>dmc1-name</u>	Name of the release 12.0 DMCL
<u>dbname or segment name</u>	Database or segment name identifying the DDLDML area to be migrated

#### 4.5.2.2 VSE/ESA execution JCL

To run RHDCMIG1 in a VSE/ESA environment in local mode, code the following execution JCL.

```
// JOB      RHDCMIG1
// LIBDEF *,SEARCH=r1201lib
// DLBL dictdb,'cdms.dictdb',da
// EXEC  PROC=IDMSLBLS
// EXEC  RHDCMIG1,SIZE=1024K
DMCL=dmcl-name
DBNAME=dbname or segment-name
/*
```

<u>IDMSLBLS</u>	Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file.
	►►For a complete listing of IDMSLBLS, see Appendix C, “IDMSLBLS Procedure for VSE/ESA JCL.”
<u>dictdb</u>	Filename of the 10.0 DDLDML area you will migrate
<u>cdms.dictdb</u>	File-ID of the 10.0 DDLDML area you will migrate area
<u>dmcl-name</u>	Name of the 12.0 DMCL
<u>dbname</u> or <u>segment-name</u>	Name of the database or segment containing the DDLDML area to be migrated

#### 4.5.2.3 VM/ESA commands

To run RHDCMIG1 in a VM/ESA environment in local mode, code the following commands.

```
FILEDEF SYSLST PRINTER
FI dictdb DISK cdms dictdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dcmmsg DISK cdms dmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FILEDEF SYSIDMS DISK sysidms input a
GLOBAL LOADLIB dbalib idmslib
OSRUN RHDCMIG1
```

<u>dictdb</u>	DDname of the 10.2 DDLDML area you will migrate. This is the ddname specified in your Release 12.0 DMCL.
<u>cdms dictdb fm</u>	File identifier of the 10.2 DDLDML area you will migrate
<u>dcmmsg</u>	DDname of the system message (DDLDCMSG) area
<u>cdms dmsgdb fm</u>	File identifier of the system message (DDLDCMSG) area

<u>ppp</u>	Page size of the database file
<u>nnn</u>	Number of pages in the database file
<u>SYSIDMS</u>	Name of the CA-supplied parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>sysidms input a</u>	<p>File identifier of the file containing the following:</p> <ul style="list-style-type: none"> <li>■ <i>dmcl-name</i> — name of Release 12.0 DMCL</li> <li>■ <i>segment-name</i> — Database name or segment name identifying the DDLDML area to be migrated</li> </ul>
<u>dbalib</u>	Filename of the load library containing the DMCL and database name table load modules for Release 12.0
<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules

#### 4.5.2.4 BS2000/OSD execution JCL

To run RHDCMIG1 in a BS2000/OSD environment in local mode, code the following execution JCL.

```

/ FILE      LINK=CDMSLIB,idms.dbalib
/ FILE      LINK=CDMSLIB1,idms.loadlib
/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSPAM,lodwork,SPACE=(primlodr,seclodr)
/ FILE      LINK=dictdb,cdms.dictdb,SHARUPD=YES
/ FILE      LINK=dcmmsg.idms.sysmsg.dd1dcmsg,SHARUPD=YES
/ FILE      LINK=SYSIDMS,*DUMMY
/ SYSFILE   SYSDTA=(SYSCMD)
/ EXEC      (RHDCMIG1,idms.loadlib)
Insert SYSIDMS parameters here. Code them
on the same or different lines.

DMCL=dmcl-name
DBNAME=dbname or segment-name

```

<u>idms.dba.loadlib</u>	Filename of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing the executable Release 12.0 modules
<u>cdms.dictdb</u>	Filename of the 10.0 DDLDML area you will migrate
<u>idms.sysmsg.dd1dcmsg</u>	Filename of the system message (DDLDMSG) area

<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>dmcl-name</u>	Name of the release 12.0 DMCL
<u>dbname or segment name</u>	Database or segment name identifying the DDLDML area to be migrated

### 4.5.3 Verify the results of RHDCMIG1

If RHDCMIG1 ran successfully, it will produce a transaction summary that:

- Lists the number of records found and modified for each record type affected by the restructure
- This message indicates it ran successfully:

```
RHDCMIG1      RELEASE 12.0
SUBS=RHDCMIGA
recname      FOUND: nnn          MODIFIED: nnn
.
.
.
END OF 12.0 MIGRATION, PHASE I
```

**Possible errors from RHDCMIG1:** If RHDCMIG1 detects a database error, all CA-IDMS/DB status fields are displayed and an abend dump taken. Note that if a database error occurs, the dictionary you are migrating will probably be corrupted (unless the error occurs during a BIND). You will have to restore the DDLDML area from a backup. Correct the problem before restarting RHDCMIG1.

**Abend code:** Possible abend code:

<b>Abend</b>	<b>Description</b>
2998	SYSLST cannot be opened. Check JCL.

### 4.5.4 Running RHDCMIG2

You may want to take a backup of the DDLDML area before running RHDCMIG2.

The execution JCL for running RHDCMIG2 in local mode is provided below.

#### 4.5.4.1 OS/390 JCL

To run RHDCMIG2 in an OS/390 environment in local mode, code the following execution JCL.

```
//RHDCMIG2 EXEC PGM=RHDCMIG2,REGION
//STEPLIB DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib,DISP=SHR
//dictdb   DD DSN=cdms.dictdb,DISP=SHR
//dcmmsg   DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR      OPTIONAL
//SYSLST   DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A      OPTIONAL
//SYSIDMS DD *               OPTIONAL

Insert SYSIDMS parameters here

DMCL=dmc1-name
DBNAME=dbname or segment name
```

<u>idms.dba.loadlib</u>	Data set name of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Data set name of the load library containing the executable Release 12.0 modules
<u>dictdb</u>	DDname of the 10.0 DDLDML area you are migrating. This is the ddname you specify in your Release 12.0 DMCL
<u>cdms.dictdb</u>	Data set name of the 10.0 DDLDML area you will migrate
<u>dcmmsg</u>	DDname of the system message (DDLDMSG) area
<u>idms.sysmsg.ddldcmsg</u>	Data set name of the system message (DDLDMSG) area
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>dmc1-name</u>	The name of a Release 12.0 DMCL
<u>dbname or segment name</u>	Database name or segment name identifying the DDLDML area to be migrated

#### 4.5.4.2 VSE/ESA JCL

To run RHDCMIG2 in a VSE/ESA environment in local mode, code the following execution JCL.

```
// JOB      RHDCMIG2
// LIBDEF *,SEARCH=R1201ib
// DLBL dictdb,'cdms.dictdb',da
// EXEC    PROC=IDMSLBLS
// EXEC    RHDCMIG2,SIZE=1024K
DMCL=dmcl-name
DBNAME=dbname or segment-name
/*
```

---

<u>IDMSLBLS</u>	Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file.  ►►For a complete listing of IDMSLBLS, see Appendix C, “IDMSLBLS Procedure for VSE/ESA JCL.”
<u>dictdb</u>	Filename of the 10.0 DDLDML area you will migrate
<u>cdms.dictdb</u>	File-ID of the 10.0 DDLDML area you will migrate area
<u>dmcl-name</u>	Name of the 12.0 DMCL
<u>dbname</u> or <u>segment-name</u>	Name of the database or segment containing the DDLDML area to be migrated

---

#### 4.5.4.3 VM/ESA commands

To run RHDCMIG2 in a VM/ESA environment in local mode, code the following commands.

```
FILEDEF SYSLST PRINTER
FI dictdb DISK cdms dictdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dmmsg DISK cdms dmmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FILEDEF SYSIDMS DISK sysidms input a
GLOBAL LOADLIB dalib idmslib
OSRUN RHDCMIG2
```

---

<u>dictdb</u>	DDname of the 10.2 DDLDML area you will migrate. This is the ddname specified in your Release 12.0 DMCL.
<u>cdms dictdb fm</u>	File identifier of the 10.2 DDLDML area you will migrate

---

<u>dcmmsg</u>	DDname of the system message (DDLDMSG) area
<u>cdms dmsgdb fm</u>	File identifier of the system message (DDLDMSG) area
<u>ppp</u>	Page size of the database file
<u>nnn</u>	Number of pages in the database file
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>sysidms input a</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ <i>dmcl-name</i> — name of Release 12.0 DMCL</li> <li>■ <i>segment-name</i> — database name or segment name identifying the DDLDML area to be migrated</li> </ul>
<u>dbalib</u>	Filename of the load library containing the DMCL and database name table load modules for Release 12.0
<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules

#### 4.5.4.4 BS2000/OSD JCL

To run RHDCMIG2 in a BS2000/OSD environment in local mode, code the following execution JCL.

```

/ FILE      LINK=CDMSLIB,idms.dbalib
/ FILE      LINK=CDMSLIB1,idms.loadlib
/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSPAM,lodrwork,SPACE=(primlodr,secloodr)
/ FILE      LINK=dictdb,cdms.dictdb,SHARUPD=YES
/ FILE      LINK=dcmmsg.idms.sysmsg.ddldcmmsg,SHARUPD=YES
/ FILE      LINK=SYSIDMS,*DUMMY
/ SYSFILE   SYSDTA=(SYSCMD)
/ EXEC      (RHDCMIG2,idms.loadlib)
Insert SYSIDMS parameters here. Code them
on the same or different lines.

DMCL=dmcl-name
DBNAME=dbname or segment-name

```

<u>idms.dba.loadlib</u>	Filename of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing the executable Release 12.0 modules

<u>cdms.dictdb</u>	Filename of the 10.0 DDLDML area you will migrate
<u>idms.sysmsg.ddldcmsg</u>	Filename of the system message (DDLDCMSG) area
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>dmcl-name</u>	Name of the release 12.0 DMCL
<u>dbname or segment name</u>	Database or segment name identifying the DDLDML area to be migrated

#### 4.5.5 Verify the results of RHDCMIG2

If RHDCMIG2 ran successfully it will produce this message:

```
RHDCMIG2 RELEASE 12.0
SUBS=IDMSNWKA
END OF 12.0 MIGRATION, PHASE II
```

**Possible errors from RHDCMIG2:** If RHDCMIG2 detects a database error, it displays all CA-IDMS/DB status fields and an abend dump is taken. Note that if a database error occurs, the dictionary you are migrating will probably be corrupted (unless the error occurs during a BIND). You will have to restore the DDLDML area from a backup. Correct the cause of the error before restarting RHDCMIG2.

Possible abend code:

Abend	Description
2998	SYSLST cannot be opened. Check JCL.

## 4.6 Run Release 12.0 IDMSDIRL utility

You must run the 12.0 IDMSDIRL utility against a migrated DDLDML area that contains the IDMSNTWK schema, if a 12.0 version of IDMSNTWK does not exist in your 12.0 environment.

The IDMSDIRL utility will delete the 10.0 IDMSNTWK schema, if present, as well as all elements and records associated with the 10.0 IDMSNTWK schema before adding the CA-supplied internal schema definitions for IDMSNTWK, IDMSSECS, and IDMSSECU, and all of the related element and record definitions and subschemas associated with these schemas.

**Note:** Only one dictionary per DC/UCF system needs to contain the IDMSNTWK schema. The SYSDIRL dictionary, created during the installation process, contains the IDMSNTWK schema.

Before you run the IDMSDIRL utility:

1. Punch the source for any user subschemas you want to save that are associated with the IDMSNTWK schema.  
IDMSDIRL will delete the 10.0 IDMSNTWK schema and all associated subschemas and add the 12.0 IDMSNTWK schema.
2. Backup the newly migrated 12.0 DDLDML area. IDMSDIRL may run longer than the dictionary migration utility. Taking a backup before you run IDMSDIRL will save considerable time if IDMSDIRL fails.

►► For guidelines on running the IDMSDIRL utility, see *CA-IDMS Utilities..*

## 4.7 Update the Task Application Table

For each application dictionary that you migrate from Release 10.2 to Release 12.0, you must update the Task Application Table (TAT) with a new 12.0 application, \$TOOLTCF. \$TOOLTCF contains the ADSA application structure for all of the Release 12.0 tools compilers (that is, ADSA, ADSC, and MAPC.) You can update the table either online using the ADSOTATU task code or in batch using the ADSOBTAT utility. \$TOOLTCF should have been linked into your 12.0 load library as part of the installation process.

- For more information on the ADSOTATU task and the ADSOBTAT utility, see *CA-ADS Reference Guide*.

Before you add the \$TOOLTCF application into your secondary dictionaries, you must delete the \$ACF@GEN application. These two applications share common task codes such as ADSA; therefore, if you add \$TOOLTCF before you remove \$ACF@GEN, an error will occur.

## 4.8 Set up 12.0 dictionary environment

This section describes how to set up your Release 12.0 dictionary environment once you have migrated your dictionaries. This includes setting up a system dictionary and your application dictionaries.

### 4.8.1 Setting up a system dictionary

**Create a separate system dictionary:** It is recommended that you separate the system dictionary from your application dictionaries; reserving it explicitly for system startup information. Maintaining a separate system dictionary, ensures that it is updated only by authorized staff in a controlled manner. This is particularly critical since the system dictionary is also required for local mode processing if CA-IDMS security is used.

**Name system dictionary SYSTEM:** You must name the dictionary DC/UCF uses to retrieve system definitions SYSTEM. This means that either the segment name of the DDLDML and DDLDCLOD areas has a name of SYSTEM or there is a database name entry of SYSTEM that includes the relevant segment, in the database name table.

**How to set up a separate system dictionary:** If your DC/UCF system definitions are currently in an application dictionary, take the following steps to move them to a separate system dictionary:

1. Define segments for the system dictionary. The SYSTEM segment should contain the DDLDML and DDLDCLOD areas. The CATSYS segment should contain the DDLCAT, DDLCATX, and DDLCATLOD areas in which your DMCL is defined.
2. Update your DMCL to include the new segments.
3. Generate, punch, and link edit the DMCL module.
4. Update your database name table to define the SYSTEM DBNAME and associate the segments defined in step 1 and the SYSMSG segment with it.
5. Format the new files that will contain the system dictionary.
6. Migrate the application dictionary that contains the system definition, if you have not done so already.
7. Punch the system definition as syntax from the migrated application dictionary.
8. Load the dictionary classes and attributes and DC device definitions into the new system dictionary using IDMSDDDL and source members DLODDEFS and DLODDCDV from the installed source library as input to IDMSDDDL.
9. Load CA-IDMS 12.0 task and program modules into the new system dictionary using IDMSDDDL and the appropriate source members from the installed source library as input to IDMSDDDL. See Chapter 5, “DC/UCF System Generation and Startup” for a list of the source library member names.
10. If you haven't already done so, load CA-IDMS 12.0 messages into the message area using IDMSDDDL and source members DLODMSG1, DLODMSG2,

DLODMSG3, DLODMSG4, DLODMSG5, and DLODMSG6 from the installed source library as input to the compiler.

11. Define your DC/UCF system in the new system dictionary using the system generation compiler and the punched system definition created in step 7. Refer to Chapter 5, “DC/UCF System Generation and Startup” to review other changes you might make to your DC/UCF system definition before generating it in a 12.0 environment.
12. Delete the DC/UCF system definition from your application dictionary.

## 4.8.2 Setting up application dictionaries

To complete the set up of your application dictionary environment, you should load required dictionary modules, records, and other components into your application dictionary. You use the DDDL compiler and the source library members listed below as input to the compiler to load these required components.

Depending on the products you have installed, some of these source library members are:

- ADSRECDS — For use with CA-ADS
- ADSRECSQ — For use with CA-ADS and SQL Option
- DLODAGN2 — For use with CA-ADS/Generator
- DLODIRPT — For use with CA-IDMS ASF and CA-ICMS
- DLODPROT — Database protocols
- DLODDEFS — Classes and attributes
- DLODAIDN — A record used to define PF or PA key values

## 4.8.3 Updating the load (DDLDCLOD) area

Depending upon the products you have installed, you should update the DDLDCLOD area with new copies of the CA-IDMS load modules provided at installation.

These modules are:

- RHDCVB — For use with CA-IDMS/DB, CA-IDMS/DC, and CA-ADS applications
- ASFIDB — For use with CA-IDMS ASF and CA-ICMS

**What's next:** The remainder of this section presents what you need to do to 10.2 schemas and subschemas to use them in a 12.0 environment.

## 4.9 Modify schemas and regenerate subschemas

**Modify schemas:** Once you have migrated DDLDML areas, you may want to modify schema source statements to take advantage of new 12.0 features. For example, you may want to use symbolic parameters. You should modify schema source statements before generating subschemas in a 12.0 environment.

If your 10.0 schemas contain record definitions that specify page range restrictions using explicit page numbers, the page numbers are converted to page offsets by the 12.0 dictionary migration process. You should evaluate the use of symbolic parameters specifying these offsets in pages or as percentages, if these records participate in a multiple dictionary environment.

If any of the schemas in your migrated dictionary are used only to define a global DMCL, that is, they *are not associated with any subschemas*, you may want to delete them from the dictionary.

**Regenerate subschemas** Whether you modify schema source statements or not, you should generate 10.0 subschemas in a 12.0 environment before using them in a 12.0 environment. You can regenerate subschemas in any of these ways::

- Use the REGENERATE ALL SUBSCHEMAS statement for each of your schemas. This is the recommended method if you want to regenerate all subschemas associated with a schema and if you want to use the new 12.0 features associated with the separation of logical and physical database definitions.

**Note:** Area page range information will be removed from all subschemas when you generate them in a 12.0 environment.

►► For a complete description of regenerating subschemas with the REGENERATE ALL SUBSCHEMAS statement, see *CA-IDMS Database Administration*.

- Use the subschema compiler to generate individual subschemas in a 12.0 environment. This method is recommended if you need to selectively regenerate subschemas.

►► For a complete discussion of regenerating subschemas using the subschema compiler, see *CA-IDMS Database Administration*.

- Use the IDMSSCON utility to convert 10.0 subschema load modules to 12.0 subschema load modules. This method is recommended for regenerating subschema load modules when the source statements don't exist.

**Note:** IDMSSCON *does not* modify the contents of subschema source. Therefore, 10.0 area page range information remains in subschemas generated using IDMSSCON.

Guidelines for using IDMSSCON are presented below.

- In a 12.0 run-time environment, when a 10.0 subschema that you do not regenerate is accessed, CA-IDMS will automatically convert the subschema to a 12.0 format. The converted subschema load module is not placed in a load area or load (core-image) library. It only exists for the duration of the run unit.

**CAUTION:**

**This is a very costly method for temporarily regenerating a subschema load module and is only provided to assist you in your conversion to Release 12.0.**

### 4.9.1 Using IDMSSCON to convert 10.0 subschema load modules

The IDMSSCON utility will convert 10.0 subschema load modules to 12.0 subschema load modules.

Here are some considerations when using IDMSSCON:

- Use IDMSSCON if you don't have subschema source statements. IDMSSCON only converts 10.0 load modules to 12.0 load modules and does not use any new 12.0 schema features.
- IDMSSCON resides in the 12.0 load (core-image) library containing CA-IDMS executable modules.
- IDMSSCON runs in local mode.
- You can convert more than one subschema load module at a time.
- 10.0 subschema load modules must be in a load (core-image) library.
- IDMSSCON will convert 10.0 subschema load modules to a 12.0 format and place them in the DDLDCLOD area of a dictionary for subsequent punching and link-editing.

### 4.9.2 Running IDMSSCON

These are the requirements for running IDMSSCON:

- You specify the 10.0 subschema name (OLD SUBSCHEMA) and the 12.0 subschema name (NEW SUBSCHEMA) as SYSPT parameters.
- The keyword OLD must begin in column 1 and the keyword NEW must begin in column 24. If you use the optional comma between old subschema name and new subschema name, it must be in column 23.
- Code each input statement on the same line
- You can convert more than one subschema at a time
- Old and new subschema names can be the same

#### 4.9.2.1 OS/390 execution JCL

To run IDMSSCON in an OS/390 environment in local mode, code the following execution JCL.

```
//IDMSSCON EXEC PGM=IDMSSCON,REGION
//STEPLIB   DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib,DISP=SHR
//          DD DSN=dbdc.r102.loadlib,DISP=SHR
//dloddb    DD DSN=idms.appldict.ddldclod,DISP=SHR
//dcmsg     DD DSN=idms.sysmsg.ddldcmmsg,DISP=SHR           OPTIONAL
//SYSLST    DD SYSOUT=A
//SYSIDMS  DD *
  
Insert SYSIDMS parameters here
DMCL=dmcl-name
DICTNAME=dbname or segment-name containing dictionary load area
//SYSIPT   DD *
  
Insert SYSIPT statements here
OLD SUBSCHEMA=10.0 subschema name,NEW SUBSCHEMA=12.0 subschema name
```

<u>idms.loadlib</u>	Data set name of the newly installed CA-IDMS/DB Release 12.0 load library containing executable Release 12.0 modules
<u>idms.dba.loadlib</u>	Data set name of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>dbdc.r102.loadlib</u>	Data set name of the load library containing 10.0 subschema load modules that you're converting with IDMSSCON
<u>dloddb</u>	DDname of the application dictionary load (DDLDCLOD) area where the converted subschema load module will be placed by IDMSSCON
<u>idms.appldict.ddldclod</u>	Dataset name of the Release 12.0 application dictionary load (DDLDCLOD) area
<u>dcmsg</u>	DDname of the system message (DDLDCMSG) area
<u>idms.sysmsg.ddldcmmsg</u>	Data set name of the system message (DDLDCMSG) area
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .

<u>dmcl name</u>	The name of the 12.0 DMCL load module to use to run IDMSSCON
<u>dictname</u>	Database name or segment name containing the DDLDCLOD area

#### 4.9.2.2 VSE/ESA execution JCL

To run IDMSSCON in a VSE/ESA environment in local mode, code the following execution JCL.

```
// JOB    IDMSSCON
// LIBDEF *,SEARCH=120 libraries and 10.2 load library
// EXEC PROC=IDMSLBL
// EXEC   IDMSSCON,SIZE=1048K
DMCL=dmcl-name
DBNAME=dbname or segment-name containing dictionary load area
/*
OLD SUBSCHEMA=10.0 subschema-name,NEW SUBSCHEMA=12.0 subschema-name
/*
```

<u>12.0 libraries and 10.2 load library</u>	Release 12.0 library and the 10.2 load library containing the subschema load module to be converted
<u>IDMSLBL</u>	Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file. ►►For a complete listing of IDMSLBL, see Appendix C, “IDMSLBL Procedure for VSE/ESA JCL.”
<u>dblod</u>	Filename of the load library containing 10.0 subschema load modules to be converted with IDMSSCON
<u>cdms.dictdb</u>	File-ID of the 10.0 DDLDML area you will migrate area
<u>dmcl name</u>	The name of the 12.0 DMCL load module to use to run IDMSSCON
<u>dictname</u>	Database name or segment name containing the DDLDCLOD area

#### 4.9.2.3 VM/ESA commands

To run IDMSSCON in a VM/ESA environment in local mode, code the following commands.

**IDMSSCON (VM/ESA)**

```

FILEDEF SYSLST PRINTER
FI dloddb DISK cdms_appllod fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dcmmsg DISK cdms_dmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FILEDEF SYSIDMS DISK sysidms input a
FILEDEF SYSIPT DISK idmsscon input a
GLOBAL LOADLIB dbalib idmslib 102idms
OSRUN IDMSSCON

```

<u>dloddb</u>	DDname of the application dictionary load (DDLDCLOD) area where the converted subschema load module will be placed by IDMSSCON
<u>cdms_appllod fm</u>	File identifier of the 12.0 application dictionary load (DDLDCLOD) area
<u>dcmmsg</u>	DDname of the system message (DDLDCMSG) area
<u>cdms_dmsgdb fm</u>	File identifier of the system message (DDLDCMSG) area
<u>ppp</u>	Page size of the database file
<u>nnn</u>	Number of pages in the database file
<u>sysidms input a</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ dmcl-name — name of Release 12.0 DMCL</li> <li>■ segment-name — database name (DBNAME) or segment name identifying the DDLDCLOD area</li> </ul>
<u>idmsscon input a</u>	File identifier of the file containing the IDMSSCON input parameters OLD SUBSCHEMA=10.2 subschema name, NEW SUBSCHEMA=12.0 subschema name
<u>dbalib</u>	Filename of the load library containing the DMCL and database name table load modules
<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules
<u>102idms</u>	Filename of the load library containing 10.2 subschema load modules that you're converting with IDMSSCON

#### 4.9.2.4 BS2000/OSD execution JCL

To run IDMSSCON in a BS2000/OSD environment in local mode, code the following execution JCL.

```

/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSLIB,idms.loadlib
/ FILE      LINK=CDMSLIB1,idms.dba.loadlib
/ FILE      LINK=CDMSLIB2,dbdc.r102.loadlib
/ FILE      LINK=dloddb,idms.appldict.ddldcload,SHARUPD=YES
/ FILE      LINK=dcmsg.idms.sysmsg.ddlcmmsg,SHARUPD=YES
/ FILE      LINK=$SIDMS,*DUMMY
/ SYSFILE   SYSDTA=(SYSCMD)
/ EXEC      (IDMSSCON,idms.loadlib)
Insert SIDMS parameters here

DMCL=dmcl-name
DBNAME=dbname or segment-name
END-SIDMS

Insert IDMSSCON parameters here

OLD SUBSCHEMA=10.0 subschema name,NEW SUBSCHEMA=12.0 subschema name

```

<u>idms.dba.loadlib</u>	Filename of the newly installed CA-IDMS/DB Release 12.0 load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing the executable Release 12.0 modules
<u>idms.appldict.ddldcload</u>	Filename of the Release 12.0 application dictionary load (DDLDCLOD) area
<u>dbdc.r102.loadlib</u>	Filename of the load library containing 10.0 subschema load modules that you're converting with IDMSSCON
<u>idms.sysmsg.ddlcmmsg</u>	Filename of the system message (DDLCMSG) area
<u>SYSIDMS</u>	Name of the parameter file where you can specify run-time directives and operating system-dependent parameters. For a complete description of the SYSIDMS parameter file, refer to <i>CA-IDMS Database Administration</i> .
<u>dmcl-name</u>	Name of the release 12.0 DMCL
<u>dbname or segment name</u>	Database or segment name identifying the DDLDML area to be migrated

### 4.9.3 Verifying results of IDMSSCON

**Using the IDMSLOOK utility:** If IDMSSCON has executed successfully, you will get a zero return code. Additionally, you may want to review a report on the contents of any of the subschema load modules IDMSSCON converts. You can use the IDMSLOOK utility program to generate a report on the contents of any subschema load module.

►► For a complete discussion of the IDMSLOOK utility, see *CA-IDMS Utilities..*

**Possible errors produced by IDMSSCON:** Here are possible errors that you could get from running the IDMSSCON utility:

- OLD SUBSCHEMA=XXXXXXXX is a required parameter starting in column 1. Correct input parameters and resubmit job.
- NEW SUBSCHEMA=XXXXXXXX is a required parameter starting in card column 24. Correct input parameters and resubmit job.
- Unable to load the old subschema from the load library specified.
- Old subschema load module is invalid.



# **Chapter 5. DC/UCF System Generation and Startup**

---

5.1 About this chapter . . . . .	5-3
5.2 About 12.0 DC/UCF system generation and start up . . . . .	5-4
5.3 Modify migrated system definition . . . . .	5-5
5.3.1 Changes made by the dictionary migration utility . . . . .	5-5
5.3.2 Other new statements and parameters . . . . .	5-9
5.3.3 Add definitions of CA-IDMS tasks and programs . . . . .	5-9
5.3.4 External user sessions . . . . .	5-11
5.3.5 Regenerate your modified system definition . . . . .	5-12
5.4 Prepare your DC/UCF environment for startup . . . . .	5-13
5.5 Review and modify startup components . . . . .	5-14



## 5.1 About this chapter

This chapter describes:

- Guidelines for reviewing your migrated system definition before you generate it in a 12.0 environment
- The changes the dictionary migration utility made to your migrated system definition
- Guidelines for modifying DC/UCF startup JCL

►►You may need to refer to *CA-IDMS System Generation* and *CA-IDMS System Operations* for information on CA-IDMS Release 12.0 system generation and start up.

## 5.2 About 12.0 DC/UCF system generation and start up

**Regenerate 10.2 system definitions:** You must generate 10.2 system definitions in a 12.0 environment before starting up a 12.0 DC/UCF system.

The dictionary migration utility made these changes to 10.2 system definitions during dictionary migration:

- Added some of the new 12.0 system generation statements
- Deleted statements and parameters no longer supported

You should review migrated system definitions before generating them in a 12.0 environment.

**Need a new 12.0 SVC:** A new CA-IDMS SVC for Release 12.0 is provided during the installation process.

**Reassemble #DCPARM macro:** You need to reassemble the #DCPARM macro before bringing up a 12.0 DC/UCF system. In addition, you may need to modify #DCPARM macro parameters.

## 5.3 Modify migrated system definition

Here are some guidelines for reviewing and modifying your migrated system definition before you generate it in a 12.0 environment:

1. Review migrated system definition
2. Review the statements and parameters the dictionary migration utility added or modified. These are listed below.
3. Change the values specified, if they don't meet your needs.
4. Verify that the information deleted from 10.2 system generation statements and parameters has been defined elsewhere. These are listed below.
5. Add new 12.0 statements or parameters that are not part of your migrated system definition. These are described below.
6. Change the SVC number to the SVC number of your new 12.0 SVC.
7. Add new CA-IDMS task and program definitions to your migrated system definition.

The tables on the next few pages list the changes the dictionary migration utility made to system definitions and identifies where system definition information that has been replaced by a 12.0 function should be specified.

### 5.3.1 Changes made by the dictionary migration utility

The dictionary migration utility made the following changes to the SYSTEM statement.

#### Deleted parameters

Parameter	Change
BLDL/NOBLDL	Not needed
LOOKAROUND TIME IS	Not needed so not replaced
PREEMPTION THRESHOLD IS	Replaced by AREA ACQUISITION THRESHOLD IS
REGISTRATION	Replaced by 12.0 RUNUNIT security
RULOCKS	Not needed so not replaced
RUNUNITS FOR EXTENT	Extent areas replaced by standard areas

**Modified parameters**

RUNUNITS FOR SIGNON/DEST	Now specified as RUNUNITS FOR SIGNON. DEST is specified on RUNUNITS FOR SYSTEM/DEST clause.
UNDEFINED PROGRAM COUNT IS FOR	If you specified ALL, each valid 10.2 program type is listed.  If you want to specify the new program type of ACCESS MODULE (SQL Option only), either <ul style="list-style-type: none"><li>■ Explicitly add ACCESS MODULE or</li><li>■ Replace existing parameters with ALL</li></ul>

**Parameters added to your system definition:** These parameters have been added to your migrated system definition:

- SYSTEM ID IS
  - Uses the nodename specified on the 10.2 DDS statement LOCAL NODE IS clause as the system ID or
  - If DDS statement is not specified, defaults to SYSTnnnn where nnnn is the value specified as *dc/ucf-version-n* on the ADD SYSTEM parameter
- AREA ACQUISITION THRESHOLD IS OFF RETRY FOREVER
- DEADLOCK DETECTION INTERVAL IS 5 <sup>1</sup>
- JOURNAL FRAGMENT INTERVAL IS 0
- JOURNAL TRANSACTION LEVEL IS 0
- RUNUNITS FOR SECURITY IS 1
- RUNUNITS FOR SYSTEM/DEST IS 1
- SCRATCH IN XA STORAGE IS NO
- XA STORAGE POOL IS 0

<sup>1</sup> The DEADLOCK DETECTION INTERVAL defaults to the value assigned to the TICKER INTERVAL

**System and teleprocessing monitor statement changes:** The dictionary migration utility made additional changes to your system definitions. These are listed in the table below. In some cases where a statement or parameter is deleted, you may need to define the information as part of another CA-IDMS component.

<b>Statement</b>	<b>Change</b>	<b>Where to redefine</b>
ADSO	Added new parameter default of OPTIONAL to the COBOL MOVE IS YES/NO clause	
DBNAME	Deleted	Use the CREATE DBTABLE statement. See <i>CA-IDMS Database Administration</i> for details on defining database name tables.
DDS	Deleted	Nodename specified on SYSTEM ID clause of the SYSTEM statement
IDMS AREA	Deleted	Specify area usage overrides using the area-override-specification clause of the ALTER DMCL statement. See <i>CA-IDMS Database Administration</i> for a complete description of physical database definition syntax.
IDMS BUFFER	Deleted	Specify buffer sizes using the BUFFER statement. See <i>CA-IDMS Database Administration</i> for a complete description of physical database definition syntax.
IDMS PROGRAM	Not used	PROGRAM REGISTRATION now specified using 12.0 run-unit security. For a complete discussion of the CA-IDMS security facility, see <i>CA-IDMS Security Administration</i> .  Specify resource limits using the TASK statement.
OLM	Set HELP PFKEY IS to PF1	

<b>Statement</b>	<b>Change</b>	<b>Where to redefine</b>
OLQ	Added ACCESS IS IDMSSQL sql COMPLIANCE IS EXTENDED	
PROGRAM	Deleted SECURITY CLASS	Define 10.2 security classes as categories using the CA-IDMS central security facility. See <i>CA-IDMS Security Administration</i> for a complete discussion of the CA-IDMS central security facility.
TASK	Deleted SECURITY CLASS and added AREA ACQUISITION THRESHOLD IS DEFAULT	Define 10.2 security classes as categories using the CA-IDMS central security facility. See <i>CA-IDMS Security Administration</i> for a complete discussion of the CA-IDMS central security facility.
USER	Removed	Define users using the CA-IDMS central security facility. See <i>CA-IDMS Security Administration</i> for details on defining users with the CA-IDMS central security facility.
LINE	Added NOPERMREADBUF (BS2000/OSD only)	
DDS LINE	Deleted statement and all PTERM parameters	Specify using the system generation NODE statement and if using CA-IDMS/DDS for cross-CPU communications, also define a CCI LINE statement.
LTERM	Deleted INTERACTIVE BREAK/NOBREAK and UPPER/LOWER parameters	Specify as profile attributes using CREATE PROFILE statement. See <i>CA-IDMS Security Administration</i> for a complete description of the CREATE PROFILE syntax.

### 5.3.2 Other new statements and parameters

Listed below are new and modified system generation statements that are not part of your migrated system definition that you may need to define.

**SYSTEM statement:** The INTERNAL WAIT parameter is not used by DC/UCF systems using CA-IDMS/DC and CICS TP monitors. It is replaced by the INACTIVE INTERVAL parameter of the SYSTEM statement. For more information, see "External request units," later in this section.

**RESOURCE TABLE statement:** You only need to use the RESOURCE TABLE statement to explicitly define remote resources that are accessed by the DC/UCF system.

**NODE statement:** You only need to use the NODE statement to define nodes on which remote resources reside.

►► For a complete description of the NODE and RESOURCE TABLE statements, see *CA-IDMS System Generation*.

**LOADLIST statement:** The version number for test load libraries is now specified as *Vnnnn* instead of *CDMSLnnn*. Remember to modify your startup JCL to reflect this change.

### 5.3.3 Add definitions of CA-IDMS tasks and programs

You need to add CA-IDMS 12.0 system generation task and program definitions to your migrated system definition to include new tasks and programs and modified task and program definitions.

**Use the INCLUDE statement:** There are several ways to accomplish this task. One of the easiest ways is to use an INCLUDE statement for each of the modules containing the task and program statements required to support the CA-IDMS products installed in your environment.

**Where to find module names:** To identify the modules you need to include in your system definition, refer to the installation job that creates the SYSTEM and SYSDIRL dictionaries, or you can refer to Appendix B in the *CA-IDMS System Generation* manual for information on CA-IDMS, CA-IDMS Tools or CA-Endevor/DB system definitions.

### System generation modules

System name	Operating System	Source library member	Module name
SYSTEM90 definition	OS/390	DLODO90	OS-SGEN90
	VSE/ESA	DLODD90	DOS-SGEN90
	BS2000/OSD	DLODB90	BS2K-SGEN90
	VM/ESA	DLODV90	CMS-SGEN90
SYSTEM99 definition	OS/390	DLODO99	OS-SGEN99
	VSE/ESA	DLODD99	DOS-SGEN99
	BS2000/OSD	DLODB99	BS2K-SGEN99
	VM/ESA	DLODV99	CMS-SGEN99

**Tasks and programs deleted:** The table below identifies the tasks and programs deleted from CA-IDMS system generation modules. These tasks and modules should be deleted from a migrated system before generation.

Source library member	Task	Program
DLODADSO	ADSG	ADSGMPDG
	ADSGT	ADSGMPDO
	ADSD	ADSGMPNC
		ADSGMPPM
		ADSGMPRS
		ADSOAMDS
		ADSOGEN1
		ADSOODSD
DLODIDDO	DMCL	IDMSDMDC
	DMCLT	IDMSDMM1
		IDMSDMTA
		IDMSDMTB
		IDMSDMTF
		IDMSDMTG
		IDMSDMTH
		IDMSDMTJ

Source library member	Task	Program
DLODO99	REPLAY	RHDCRPLY
	DCPASS	IDMSNWKC
		IDMSNWKM
		IDMSNWKN
		IDMSNWKO
		IDMSNWKP
		IDMSNWKQ
		IDMSNWKS
		IDMSNWKX
		IDMSWK1
		IDMSWK2
		IDMSWK4
		IDMSSPF
		RHDCPASS

### 5.3.4 External user sessions

**What is an external user session:** An external user session is a logical connection between a DC/UCF system and an application executing outside that DC/UCF system. There are different types of external user sessions; each using a different communication method to connect the application to a DC/UCF system. External user sessions can be:

- External request units (ERUs)
- Applications using CAICCI to communicate with DC/UCF such as CA-IDMS/DDS and CA-Visual Express
- LU 6.2 sessions

**What is an external request unit:** An external request unit is a request for DC/UCF system services which originates outside the DC/UCF address space and uses the SVC to communicate between the application and the DC/UCF system.

There are three types of external request units:

- CA-IDMS batch and CICS applications requesting database services
- Application programs requesting DC/UCF services using CA-IDMS/UCF front-end modules
- CA-IDMS/DC applications requesting DC or database services from a different address space within the same mainframe

In CA-IDMS 12.0, all requests for IDMS services from other address spaces are processed and controlled as online tasks (that is, as if they were initiated from within the same address space). The next section highlights system generation statements that you might need to modify to manage some types of external user sessions in 12.0.

**Specifying MAXIMUM ERUS parameter:** Use the SYSTEM statement MAXIMUM ERUS parameter to specify the maximum number of concurrent external user sessions that can be serviced by a DC/UCF system.

**Specifying execution characteristics:** Characteristics of external run units were specified in 10.2 by using an IDMS PROGRAM statement. In 12.0, you specify them on the TASK statement. TASK statement parameters are used to manage external request units (except applications using UCF front-end modules) and external user sessions using CAICCI, such as CA-IDMS/DDS and CA-IDMS/QbyX. The correlation between the 10.2 IDMS PROGRAM statement and the 12.0 TASK statement is summarized below.

IDMS PROGRAM statement parameters	TASK statement parameters
EXTERNAL WAIT TIME	EXTERNAL WAIT TIME
INTERNAL WAIT TIME	INACTIVE INTERVAL
LIMITS	LIMITS
PREEMPTION THRESHOLD	AREA ACQUISITION THRESHOLD
PRIORITY	PRIORITY
STATUS	ENABLED/DISABLED

Program registration and the SUBSCHEMA clause of the IDMS PROGRAM statement are replaced by 12.0 run-unit security.

**The RHDCNP3S task:** You can specify execution characteristics for non-UCF external request units and CAICCI sessions by using either the RHDCNP3S task, or you can override RHDCNP3S by defining a task code for a front-end environment (batch or CICS) or a specific application.

A sample RHDCNP3S task definition is provided at installation.

►►For more information on specifying run-time characteristics for external user sessions, see *CA-IDMS System Generation* and *CA-IDMS System Operations*.

### 5.3.5 Regenerate your modified system definition

Once you have reviewed and modified your DC/UCF system definition, generate it in a 12.0 environment.

## 5.4 Prepare your DC/UCF environment for startup

Before you start up your 12.0 DC/UCF environment, perform these tasks:

1. Format DDLDCCLOG and DDLDCCRUN areas.

The format of some of the records in the DDLDCCLOG and DDLDCCRUN areas are different.

2. Format DDLDCCSCR area.

3. Format journal files.

Journal record formats have changed.

4. Prepare ARCHIVE JOURNAL, ARCHIVE LOG, and PRINT LOG statements.

IDMSAJNL utility program is replaced by the ARCHIVE JOURNAL utility statement and the RHDCPRLG utility program is replaced by the new utility statements ARCHIVE LOG and PRINT LOG.

You should prepare these new utility statements and JCL streams to replace IDMSAJNL and RHDCPRLG utility programs before bringing up your 12.0 system. This is especially important if you use a write-to-operator (WTOEXIT) exit to automatically initiate archive journal and archive and print log functions.

- For a detailed description of all utility statements and programs, see *CA-IDMS Utilities*.

## 5.5 Review and modify startup components

Review the list of system startup components below and make the necessary changes in your environment before bringing up your 12.0 DC/UCF system.

- For a complete description of DC/UCF system startup procedures, see *CA-IDMS System Operations*.

**Reassemble the #DCPARM macro:** You must reassemble the #DCPARM macro whether or not you change any of the parameter values.

### System startup components

Item	Action
#DCPARM macro	Review and modify the #DCPARM macro to include any changed parameters such as: <ul style="list-style-type: none"><li>■ Global DMCL name</li><li>■ FREESTG</li><li>■ DC/UCF system version number</li></ul>
WTOEXIT	Review the WTOEXIT source code to determine if any changes are needed to accommodate 12.0 control block changes.
Link edit the #DCPARM and WTOEXIT object modules	Link edit the #DCPARM and WTOEXIT object modules with the appropriate operating system-dependent module to create a new startup module
Startup routine JCL	Modify startup routine JCL to include changes to your environment for Release 12.0 such as: <ul style="list-style-type: none"><li>■ New startup module</li><li>■ Region size</li><li>■ New data set names of the required CA-IDMS libraries</li><li>■ New data set names of migrated dictionaries</li><li>■ New data set names of additional dictionary areas and user catalog</li><li>■ Replace CDMSL<math>nnn</math> names with V<math>nnnn</math></li><li>■ Any other required changes</li></ul>

# **Chapter 6. Utilities**

---

6.1 About this chapter . . . . .	6-3
6.2 Utility statements . . . . .	6-4
6.3 Utility programs . . . . .	6-6
6.4 Security for utilities . . . . .	6-8



## 6.1 About this chapter

This chapter lists the new CA-IDMS utility statements and identifies the 10.2 utility programs they replace. Enhancements to the remaining utility programs are presented.

## 6.2 Utility statements

**Utility statements that replace programs:** The following table shows the new utility statements that replace existing utility programs.

Program	New utility statement
IDMSAJNL	ARCHIVE JOURNAL
IDMSDBLU	FASTLOAD
	RELOAD
IDMSDUMP	BACKUP
	PRINT SPACE
IDMSINIT	FORMAT
IDMSJFIX	PRINT JOURNAL
	FIX ARCHIVE
IDMSLDEL	CLEANUP
IDMSPCON	RESTRUCTURE CONNECT
IDMSPFIX	FIX PAGE
	PRINT PAGE
	UNLOCK
IDMSPTRE	PRINT INDEX
IDMSRBCK	ROLLBACK
IDMSRFWD	ROLLFORWARD
IDMSRSTR	RESTORE
IDMSRSTU	RESTRUCTURE SEGMENT
IDMSTBLU	MAINTAIN INDEX
	MAINTAIN ASF TABLE
IDMSUNLD	UNLOAD
IDMSXPAG	EXPAND PAGE
RHDCPRLG	ARCHIVE LOG
	PRINT LOG

**CAUTION:**

The format of the files created by IDMSDUMP is not compatible with the format expected by RESTORE. RESTORE accepts only files created by BACKUP.

Therefore, once you have converted your environment, you should backup your database using BACKUP so you have a backup file that RESTORE can use, if you need to restore your database.

**PUNCH statement:** In addition to the other statements that replace programs, there is a new utility statement called PUNCH. PUNCH:

1. Retrieves DMCL load modules or database name table load modules from the catalog load area of the data dictionary
2. Writes the load modules in object form to the SYSPCH file

**Note:** When you use the PUNCH statement with the CA-IDMS Batch Command Facility, you can only punch one component at a time. You need to create separate JCL streams for each PUNCH statement.

**Execution JCL:** You need to create new execution JCL to run the new utility statements. You submit utility statements to the CA-IDMS Batch Command Facility (IDMSBCF).

- For an overview of the basic execution JCL for the CA-IDMS Batch Command Facility, see *CA-IDMS Command Facility*.
- For a complete description of the execution JCL for each utility statement, see *CA-IDMS Utilities*.

## 6.3 Utility programs

**Status of utility programs:** These utilities exist in their previous format:

IDMSCALC  
IDMSDBAN (with enhancements)  
IDMSDIRL (with enhancements)  
IDMSLOOK (with enhancements)  
IDMSRADM  
IDMSRPTS (with enhancements)  
IDMSRSTC

**Changes in existing programs:** The following table describes enhancements to utilities that continue to exist in their previous format.

Utility	Enhancements
IDMSDBAN	New syntax Support for SQL-defined databases Audits indexes more efficiently
IDMSDIRL	Loads IDMSNTWK schema and two associated subschemas (IDMSNWKA and IDMSNWKG) Loads schemas IDMSSECS and IDMSSECU and subschemas IDMSSECS and IDMSSECU used for security processing Added option (SCHEMA-DELETE) to remove all CA-defined schema definitions (those listed above) from a dictionary without loading new definitions

Utility	Enhancements
IDMSLOOK	<p>Revised parameters provide this output:</p> <ul style="list-style-type: none"> <li>▪ Set information (SUBSCHEMA)</li> <li>▪ Buffer information (DMCL)</li> <li>▪ Information reflecting the separation of logical and physical definitions (DBTABLE)</li> <li>▪ Information on segments (DBTABLE)</li> <li>▪ Data set name of the library that the load module was loaded from (DATES and PROGRAM)</li> </ul> <p>New parameters provide:</p> <ul style="list-style-type: none"> <li>▪ Information about IDMSLOOK parameters (HELP)</li> <li>▪ Information about subschema load modules bound to a database (BIND SUBSCHEMA)</li> <li>▪ Value of a date/time stamp (DATETIME STAMP)</li> <li>▪ Additional parameters for SQL-related objects</li> </ul>
IDMSRPTS	<ul style="list-style-type: none"> <li>▪ Removal of physical information from schema and subschema reports</li> <li>▪ Enhancements to schema and subschema reports — Reporting on new compiler functions</li> <li>▪ New reports for physical database definitions</li> </ul>

**Removal of IDMSRNWK:** IDMSRNWK no longer exists. Because of the separation of logical and physical definitions, there is no longer a need for dictionary subschema reformatting.

**Local mode execution JCL:** You need to modify the execution JCL to run utility programs in local mode. You use the SYSIDMS parameter file to identify the DMCL, dictionary, or database CA-IDMS/DB will use to execute the utility program in local mode.

Additionally, you may use the SYSIDMS parameter file to specify other parameters, depending upon the utility program.

►► For a complete description of utility programs and associated execution JCL, see *CA-IDMS Utilities*.

## 6.4 Security for utilities

The execution of utility statements is subject to security checking. The specific authority necessary to execute a utility statement depends on the function of the utility.

*CA-IDMS Utilities* describes the authority required to use each utility statement and program.

# Chapter 7. User Exits and Database Procedures

---

7.1 About this chapter . . . . .	7-3
7.2 User exits . . . . .	7-4
7.2.1 New user exits . . . . .	7-4
7.3 Database procedures . . . . .	7-5



## 7.1 About this chapter

This chapter identifies the changes you need to make to CA-IDMS user exits and database procedures in order to use them in a 12.0 environment.

## 7.2 User exits

In Release 12.0, new user exits have been added and one user exit has been deleted. Additionally, the CA-IDMS/DB architecture has changed and the format of control blocks is different.

### 7.2.1 New user exits

Here is a list of the new user exits:

- Exit 28 — Security preprocessing
- Exit 29 — Security postprocessing
- Exit 30 — Deadlock victim selection
- Exit 31 — Transaction statistics

**User exit 3 is replaced:** Release 10.2 user exit 3, the security check exit, is replaced by the new user exits 28 and 29. If you will continue to perform security checking in the 12.0 environment, use exits 28 and 29.

**Control block changes:** Because the CA-IDMS/DB database architecture has changed significantly, the format of control blocks is different.

Review the user exits you use in your CA-IDMS environment carefully to determine the impact of control block changes.

You should make the necessary changes to the exit and then reassemble or recompile and relink each user exit in the 12.0 environment.

## 7.3 Database procedures

The control blocks passed to database procedures have changed:

- The names of records, areas, error sets, error areas, etc. are now 18 bytes
- The order of the record control block is different

You must modify your CA-IDMS database procedures to reflect control block changes.



# Chapter 8. Extent Areas

---

8.1 About this chapter . . . . .	8-3
8.2 Migrate ASF extent areas . . . . .	8-4
8.2.1 Prepare your environment to migrate extent areas . . . . .	8-4
8.2.2 Run IDMSRSSM program . . . . .	8-5
8.2.2.1 Execution JCL . . . . .	8-6
8.2.3 Output from IDMSRSSM . . . . .	8-9
8.2.4 Generate the IDMSRSSD subschema . . . . .	8-13
8.2.5 Unload and reload ASF data areas . . . . .	8-13
8.2.6 Unload and reload ASF definition area . . . . .	8-14
8.2.7 Run IDMSRUPD program . . . . .	8-14
8.2.8 Execution JCL . . . . .	8-15
8.2.9 Output from IDMSRUPD . . . . .	8-18
8.2.10 Optionally, add CALC record to IDMSR schema . . . . .	8-20
8.2.11 Execution JCL . . . . .	8-20



## 8.1 About this chapter

Extent areas are not supported in Release 12.0.

**DDLDCQUE area:** The DDLDCQUE area is not used in the CA-IDMS 12.0 environment.

Use the DDLDCCRUN area in place of the DDLDCQUE area. Remove any references you may have to the DDLDCQUE area in physical database definition components or application programs.

**ASF extent areas:** You must migrate ASF extent areas to standard areas in a 12.0 environment. This chapter describes how to migrate ASF extent areas to standard areas.

## 8.2 Migrate ASF extent areas

CA-IDMS provides two programs, along with standard CA-IDMS utility statements, to assist you in migrating ASF extent areas to standard areas.

**Procedures:** Migrating ASF extent areas involves these steps:

1. Prepare environment to migrate extent areas.
2. Run the IDMSRSSM program to:
  - Create syntax for the IDMSRSSD subschema that you will use to unload and reload ASF data areas defined as extent areas.
  - Update the catalog structure of the ASF dictionary with revised storage allocation information.
3. Review IDMSRSSD subschema syntax produced by the IDMSRSSM program and then generate it.
4. Unload and reload ASF data areas defined as extent areas.
5. Unload and reload the ASF definition area.
6. Run the IDMSRUPD program to update the RDEFREC records in the ASF definition area with the new db-keys of the table header records (RFUR-nnnnnn-OOAK).
7. Optionally, add a CALC record to the IDMSR schema to define an SR1 system record for 12.0 processing of ASF tables.

Each of these steps is described in the remainder of this chapter.

### 8.2.1 Prepare your environment to migrate extent areas

**Release 10.2 environment:** Before you migrate any ASF extent areas, take these steps in your 10.2 environment:

1. Identify the extent areas and the ASF-defined tables you want to migrate. Make sure the tables you will migrate are completely generated. The migration program will not migrate a table that is not completely generated or does not have a validated subschema.

The IDMSRSSM program will identify all tables which cannot be migrated. If you want to migrate such a table, you must first generate (or regenerate) the table using 10.2 ASF.

2. Run the 10.2 IDMSDUMP utility program with the REPORTS=ONLY option against each ASF extent area (data areas and definition area) you will migrate. You can compare the records listed on the IDMSDUMP report with the output listing from the IDMSRSSM program to determine if all tables are processed by IDMSRSSM.
3. Backup your 10.2 format ASF dictionary and extent areas. You will need these if you must generate or regenerate an ASF table using 10.2 ASF.

**Release 12.0 environment:** Before you begin the migration process, be sure your Release 12.0 environment is prepared as follows:

1. A 12.0 DMCL contains all ASF areas you will migrate as well as ASF dictionary areas.

**Tip:** You could create a segment that contains all ASF areas being migrated or add ASF areas to an existing segment. If you make either of these changes, be sure your 12.0 DMCL contains the change.

**Note:** ASF areas are defined as standard areas in your 12.0 DMCL.

2. ASF dictionaries are in a 12.0 format (i.e. they must have been migrated to a 12.0 format using the Release 12.0 Dictionary Migration Utility).

## 8.2.2 Run IDMSRSSM program

**Purpose:** Once your environment is prepared, run the IDMSRSSM program separately for each migrated ASF dictionary on a 12.0 system. The IDMSRSSM program:

- Creates the subschema syntax for the IDMSRSSD subschema and places it in an output file. The IDMSRSSD subschema is used to unload and reload ASF data areas.
- Updates the catalog structure in the ASF dictionary with the maximum and current table allocations for each user. Space allocations are now expressed as a number of tables rather than in K bytes.

**Important:** In a 10.2 environment, if the user defaults total allocation limit is 1000 K bytes, IDMSRSSM converts that to a total allocation limit of 1000 tables for Release 12.0. You should review these allocations and modify or add them as appropriate.

►► For information on defining and modifying allocation limits and defaults for user's tables, see *CA-IDMS ASF User Guide*.

- Indicates whether you need to add a CALC record to the IDMSR schema so that the SR1 system record is included in the schema for 12.0 ASF processing.
- Produces a listing of record IDs for the tables in the IDMSR schema version 1.

### CAUTION:

**Tables that don't have an existing or validated subschema are not included in the IDMSRSSD subschema.**

You can run the IDMSRSSM program against the same ASF dictionary area as often as is necessary. However, only the initial run of IDMSRSSM will update the catalog structure of the ASF dictionary. Subsequent executions of IDMSRSSM will not perform any updates but will report on the allocations.

### Syntax

►►↓ - AREA=extent-area-name ►►

**Parameter****AREA=extent-area-name**

Specifies the name of the ASF extent area you will migrate. *Extent-area-name* must be an extent area defined in the dictionary.

You can specify a maximum of 20 area statements for each run of the IDMSRSSM program.

**8.2.2.1 Execution JCL**

IDMSRSSM is a program in the load library provided at installation.

IDMSRSSM runs in a batch environment in either local mode or under the central version.

Execution JCL for the IDMSRSSM program is provided below.

**OS/390 JCL: IDMSRSSM**

```
//BUILD EXEC PGM=IDMSRSSM,REGION=
//STEPLIB DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib.,DISP=SHR
//asfdict DD DSN=idms.asfdict.ddldm1,DISP=SHR
//dcmsg DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR
//sysjrn1 DD DSN=idms.tapejrn1,DISP=(NEW,KEEP),UNIT=TAPE
//SYS004 DD DSN=output.file,DISP=(,CATLG),UNIT=
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=nnnn)
//SYSLST DD SYSOUT=A
//SYSIDMS DD *
DMCL=dmcl-name
DBNAME=dbname or segment-name
//SYSIPT DD *
AREA=extent-area-name     AREA=extent-area-name
```

<u>idms.dba.loadlib</u>	Data set name of the load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Data set name of the load library containing executable Release 12.0 CA-IDMS modules
<u>asfdict</u>	DDname of the ASF (DDLDML) dictionary area
<u>idms.asfdict.ddldm1</u>	Data set name of the ASF (DDLDML) dictionary area
<u>dcmsg</u>	DDname of the system message (DDLDCMSG) area
<u>idms.sysmsg.ddldcmsg</u>	Data set name of the system message (DDLDCMSG) area
<u>sysjrn1</u>	DDname of a tape journal file, if journaling
<u>idms.tapejrn1</u>	Dataset name of tape journal file

---

<u>sys004</u>	DDname of the output file for the subschema produced by the IDMSRSSM program
<u>output.file</u>	Data set name of the output file for the subschema produced by the IDMSRSSM program
<u>dmcl-name</u>	The name of a Release 12.0 DMCL
<u>dbname or segment-name</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>extent-area-name</u>	The name of an ASF extent area

---

To run IDMSRSSM under the central version, add a SYSCTL file.

#### VSE/ESA JCL: IDMSRSSM

```
// JOB IDMSRSSM
// LIBDEF *,SEARCH=120 libraries
// EXEC PROC=IDMSLBLS
// ASSGN sysnnn,DISK,VOL=nnnnnn,SHR
// DLBL SYS004,'output.file'
// EXEC IDMSRSSM,SIZE=1048K
DMCL=dmcl-name
DBNAME=dbname or segment-name
/*
AREA=extent-area-name    AREA=extent-area-name
/*
```

---

<u>IDMSLBLS</u>	Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file.
	►►For a complete listing of IDMSLBLS, see Appendix C, "IDMSLBLS Procedure for VSE/ESA JCL."
<u>sys004</u>	Filename of the output file for the subschema produced by IDMSRSSM program
<u>dmcl-name</u>	The name of a Release 12.0 DMCL
<u>dbname or segment-name</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>extent-area-name</u>	The name of an ASF extent area

---

To run IDMSRSSM under the central version, add a SYSCTL file.

**VM/ESA commands: IDMSRSSM**

```

FILEDEF SYSLST PRINTER
FI asfdict DISK cdms asfdict fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dcmmsg DISK cdms dmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FILEDEF sysjrnl TAPI SL VOLID nnnnnn (RECFM VB LRECL lll BLKSIZE bbbb
FILEDEF SYS004 DISK sys004 output a
FILEDEF SYSIDMS DISK sysidms input a
FILEDEF SYSIPT DISK idmsrssm input a
GLOBAL LOADLIB dbalib idmslib
OSRUN IDMSRSSM

```

<u>asfdict</u>	DDname of the ASF (DDLDML) dictionary area
<u>cdms asfdict fm</u>	File identifier of the ASF (DDLDML) dictionary area
<u>dcmmsg</u>	DDname of the system message (DDLDCMSG) area
<u>cdms dmsgdb fm</u>	File identifier of the system message (DDLDCMSG) area
<u>ppp</u>	Page size of the database file
<u>nnn</u>	Number of pages in the database file
<u>sysjrnl</u>	DDname of the tape journal file, if journaling
<u>nnnnnn</u>	Volume serial number of the tape journal file
<u>lll</u>	Record length of the tape journal file
<u>bbbb</u>	Block size of the tape journal file
<u>sys004 output a</u>	File identifier of file containing subschema produced by IDMSRSSM program
<u>sysidms input a</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ dmcl-name — Name of the DMCL module</li> <li>■ segment-name — database name or segment name identifying the DDLDML area of the ASF dictionary</li> </ul>
<u>idmsrssm input a</u>	File identifier of the file containing the names of the ASF extent areas you will migrate
<u>dbalib</u>	Filename of the load library containing the DMCL and database name table load modules
<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules

**BS2000/OSD JCL: IDMSRSSM**

```

/ FILE      LINK=CDMSLIB,idms.dbalib
/ FILE      LINK=CDMSLIB1,idms.loadlib
/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSPAM,1odrwork,SPACE=(primladr,secladr)
/ FILE      LINK=asfdict,idms.asfdict.ddldml,SHARUPD=YES
/ FILE      LINK=dcmsg,idms.sysmsg.ddldcmsg,SHARUPD=YES
/ FILE      LINK=sysjrn1,idms.tapejrn1
/ FILE      LINK=SYS004,output.file
/ FILE      LINK=SYSIDMS,*DUMMY
/ SYSFILE   SYSDTA=(SYSCMD)
/ EXEC      (IDMSRSSM,idms.loadlib)

Insert SYSIDMS parameters here

DMCL=dmcl-name
DBNAME=dbname or segment-name
END-SYSIDMS

Insert IDMSRSSM parameters here

AREA=extent-area-name          AREA=extent-area-name

```

<u>idms.dba.loadlib</u>	Filename of the load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules
<u>idms.asfdict.ddldml</u>	Filename of the ASF (DDLDML) dictionary area
<u>idms.sysmsg.ddldcmsg</u>	Filename of the system message (DDLDMSG) area
<u>idms.tapejrn1</u>	Filename of tape journal file
<u>output.file</u>	Filename of the output file for the subschema produced by the IDMSRSSM program
<u>dmcl-name</u>	The name of a Release 12.0 DMCL
<u>dbname or segment-name</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>extent-area-name</u>	The name of an ASF extent area

To run IDMSRSSM under the central version, add a SYSCTL file.

### 8.2.3 Output from IDMSRSSM

#### What IDMSRSSM produces

- An output listing that includes various information. This information is described in the table below.
- Subschema syntax for the IDMSRSSD subschema.

## 8.2 Migrate ASF extent areas

---

**Sample IDMSRSSM output listing:** Portions of a sample output listing from IDMSRSSM are provided below.

Allocation limits are represented as tables instead of K bytes and allocation defaults are removed from the catalog.

```
RECORD ID: 31839
RECORD ID: 31840
RECORD ID: 31841
RECORD ID: 31842
```

```
VALIDATE NOT DONE FOR SUBSCHEMA:
RECORD: RFUR-000217-00AK          WILL NOT BE INCLUDED
IN SYNTAX FOR AREA: IDMSR-AREA2

RECORD ID: 31913
VALIDATE NOT DONE FOR SUBSCHEMA:
RECORD: RFUR-000217-DATA          WILL NOT BE INCLUDED
IN SYNTAX FOR AREA: IDMSR-AREA2

RECORD ID: 31914
VALIDATE NOT DONE FOR SUBSCHEMA:
RECORD: RFUR-000218-00AK          WILL NOT BE INCLUDED
IN SYNTAX FOR AREA: IDMSR-AREA2
```

```
RECORD ID: 31934
RECORD: RFOR-000135-00AK          WILL NOT BE INCLUDED
IN SYNTAX FOR AREA: IDMSR-AREA2

RECORD ID: 31935
RECORD: RFOR-000135-DATA          WILL NOT BE INCLUDED
IN SYNTAX FOR AREA: IDMSR-AREA2

RECORD ID: 31936
```

```
RECORD ID: 31998
RECORD ID: 31999
AREAS ADDED: 0002 RECORDS ADDED: 0140 SETS ADDED: 0124

UPDATING CATALOG

USER: CORP                      EXG
VALUES FOUND: TOTAL USED: 00002291 COUNT OF BYTES: 00002291
REPLACED WITH TABLE COUNT: 00010
COUNT OF TABLES UNALLOCATED: 00001

USER: CORP                      W FURR
VALUES FOUND: TOTAL USED: 00000000 COUNT OF BYTES: 00000000
REPLACED WITH TABLE COUNT: 00000
```

```

USER: CORP          DNT
VALUES FOUND: TOTAL USED: 00001014   COUNT OF BYTES: 00001014
REPLACED WITH TABLE COUNT: 00004
COUNT OF TABLES UNALLOCATED: 00004

USER: CORP          SJG
USER DEFAULTS TOTAL ALLOCATION: 00000100
BEING REMOVED - PRIMARY: 00000 SECONDARY: 00000 MAXIMUM: 00000
VALUES FOUND: TOTAL USED: 00030468   COUNT OF BYTES: 00030968
REPLACED WITH TABLE COUNT: 00011

USER: CORP          CULL DBA
VALUES FOUND: TOTAL USED: 00001512   COUNT OF BYTES: 00001513
REPLACED WITH TABLE COUNT: 00002
COUNT OF TABLES UNALLOCATED: 00001

COUNT OF USERS UPDATED: 00025

```

**Review the IDMSRSSM output listing:** The table below describes the information provided on the listing from IDMSRSSM that you should review. You should review this information as it might require you to take some action. Suggested actions are provided to assist you.

If you need to change the status of any ASF table after reviewing the output listing:

- Make the changes using 10.2 ASF on a 10.2 system using the backup copy of the 10.2 ASF dictionary
- Migrate the 10.2 dictionary to a 12.0 format
  - For complete instructions on migrating a dictionary to a 12.0 format, see Chapter 3, “Security Migration.”
- Rerun IDMSRSSM to reflect the changes

Item in report	What it means	What to do
Optionally, this message:  A DUMMY CALC RECORD MUST BE ADDED	This message appears after the area statements at the beginning of the listing only if an SR1 record needs to be added to the IDMSR schema.	If this message appears on your listing, add a record with a location mode of CALC to your IDMSR schema. See 8.2.10, “Optionally, add CALC record to IDMSR schema” on page 8-20 for instructions on how to include a CALC record in the IDMSR schema.
RECORD ID	The record ID of a table included in the IDMSRSSD subschema.	If you produced an IDMSDUMP report, compare it with the list of record IDs produced by IDMSRSSM to determine if all tables were processed.

<b>Item in report</b>	<b>What it means</b>	<b>What to do</b>
VALIDATE NOT DONE FOR SUBSCHEMA .....	Identifies the record ID of a table that does not have a validated subschema.	If you want to migrate this table, use ASF on a 10.2 system to determine why the subschema is not validated, fix the problem, and generate the table.
RECORD: RFOR-nnnnnn-DATA WILL NOT BE INCLUDED IN SYNTAX.....	Identifies tables that have existing definition record names preceded with 'RFOR-'. Existing record names preceded with 'RFOR-' indicate that a table is not completely regenerated. These records are not included in the IDMSRSSD subschema.	If you want to migrate the data from tables that correspond to an existing 'RFOR' record, access the table through ASF on a 10.2 system and complete the regeneration process.
TOTAL USED	The total number of K bytes a user's tables consume.	
COUNT OF BYTES	The number of K bytes allocated for a user as calculated by IDMSRSSM.	
REPLACED WITH TABLE COUNT	The number of tables generated by a user. This number is calculated by IDMSRSSM from information in the catalog and is now maintained as a number of tables instead of K bytes.	
COUNT OF TABLES UNALLOCATED	The number of tables defined but not generated by a user.	

Item in report	What it means	What to do
USER DEFAULTS TOTAL ALLOCATION	The maximum number of tables a user can define. This was previously maintained in K bytes and is now represented as a number of tables. For example, if the user defaults total allocation for user PRK was 100 K bytes in 10.2, IDMSRSSM changes that to 100 tables in 12.0.	Review the updated allocation limits. If necessary, use ASF on a 12.0 system to change them to meet the requirements of your environment.
BEING REMOVED - PRIMARY: SECONDARY: MAXIMUM	The values associated with these allocation defaults are removed from the catalog and are no longer maintained.	

### 8.2.4 Generate the IDMSRSSD subschema

After reviewing the IDMSRSSM output, use the subschema compiler in a 12.0 environment to generate the IDMSRSSD subschema.

**Note:** IDMSRSSD is the default subschema name used by the IDMSRUPD program if a subschema name is not provided. You can change the subschema name before generating it. If you do, be sure to specify the new name when running the UNLOAD and RELOAD utility statements against ASF data areas and the IDMSRUPD program.

### 8.2.5 Unload and reload ASF data areas

After you generate the IDMSRSSD subschema, you need to:

1. Unload ASF data areas using the 12.0 UNLOAD utility statement.
2. Format each data area using the 12.0 FORMAT utility statement.
3. Reload the data unloaded by the previous unload step into the formatted 12.0 area using the 12.0 RELOAD utility statement.

►For a complete description of 12.0 utility statements, see *CA-IDMS Utilities*.

## 8.2.6 Unload and reload ASF definition area

After you unload and reload ASF data areas, you can unload and reload the ASF definition area using the IDMSRSSA subschema. The IDMSRSSA subschema is only installed with ASF.

**Note:** All 10.2 subschemas must be in a 12.0 format for use in a 12.0 environment. CA-IDMS automatically converts a 10.2 subschema load module to a 12.0 format (if it isn't in a 12.0 format) when it accesses it at runtime. The converted 12.0 subschema load module is temporary and is not saved in a load area or load library. It only exists for the duration of the run unit. It is recommended that you permanently convert your 10.2 subschemas to a 12.0 format. For procedures on permanently converting a 10.2 subschema to a 12.0 subschema, see Chapter 3, "Security Migration."

1. Unload ASF definition area using the 12.0 UNLOAD utility statement
2. Format the definition area using the 12.0 FORMAT utility statement
3. Reload the data unloaded by the previous unload step into the formatted 12.0 definition area using the 12.0 RELOAD utility statement

►►For a complete description of 12.0 utility statements, see *CA-IDMS Utilities*.

## 8.2.7 Run IDMSRUPD program

Once the ASF definition area and ASF data areas are migrated, you must run the IDMSRUPD program to update RDEFREC records in the ASF definition area with the new db-keys of related table header records (RFUR-nnnnnn-OOAK).

**Purpose:** The IDMSRUPD program:

- Accesses each RFUR-nnnnnn-OOAK record in the subschema produced by the IDMSRSSM program (default subschema name is IDMSRSSD) to get the table definition number of the table it represents
- Accesses the RDEFREC that the table definition number corresponds to in the definition area and updates the RDEFREC with the db-key of its corresponding RFUR-nnnnnn-OOAK record

### Syntax

►► [ SUB=subschema-name ] [ SUB=IDMSRSSD ] ►►

### Parameters

#### **SUB=subschema-name**

Specifies the name of the subschema the IDMSRUPD program will use.  
*Subschema-name* must be a 1- through 8- character alphanumeric value.

If you do not specify a *subschema-name*, IDMSRUPD will use IDMSRSSD as the default subschema name.

## 8.2.8 Execution JCL

The IDMSRUPD program runs in a batch environment in local mode.

**Note:** Depending on the size of your data areas, IDMSRUPD may be a long running job.

Execution JCL for IDMSRUPD is provided below.

**OS/390 JCL:** To run IDMSRUPD in local mode, code this JCL.

### IDMSRUPD

```
//BUILD EXEC PGM=IDMSRUPD,REGION=
//STEPLIB DD DSN=idms.dba.loadlib,DISP=SHR
//          DD DSN=idms.loadlib,DISP=SHR
//asfdef DD DSN=idms.asfdef,DISP=SHR

If processing more than one data area,
add dd statements for them.

//asfdata DD DSN=idms.asfdata,DISP=SHR
//dcmsg DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR
//sysjrn1 DD DSN=idms.tapejrn1,DISP=(NEW,KEEP),UNIT=TAPE
//SYSLST DD SYSOUT=A
//SYSIDMS DD *

DMCL=dmc1-name
DBNAME=segment-name

//SYSIPT DD *

SUB=subschema-name
```

<u>idms.dba.loadlib</u>	Data set name of the load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Data set name of the load library containing executable Release 12.0 CA-IDMS modules
<u>asfdef</u>	DDname of the ASF definition area
<u>idms.asfdef</u>	Data set name of the ASF definition area
<u>asfdata</u>	DDname of an ASF data area
<u>idms.asfdata</u>	Data set name of the ASF (DDLML) dictionary area
<u>dcmsg</u>	DDname of the system message (DDLCMSG) area
<u>idms.sysmsg.ddldcmsg</u>	Data set name of the system message (DDLCMSG) area
<u>sysjrn1</u>	DDname of the tape journal file, if journaling
<u>idms.tapejrn1</u>	Dataset name of the tape journal file, if journaling

<u>dmcl-name</u>	The name of a Release 12.0 DMCL
<u>dbname</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>subschema-name</u>	Name of the subschema IDMSRUPD will use if not IDMSRSSD

### VSE/ESA JCL IDMSRUPD

```
// JOB    IDMSRUPD
// LIBDEF *,SEARCH=120 libraries
// EXEC PROC=IDMSLBL
// ASSGN sysnnn,DISK,VOL=nnnnnn,SHR
// EXEC   IDMSRUPD,SIZE=1048K
DMCL=dmcl-name
DBNAME=dbname or segment-name
/*
SUB=subschema-name
/*
```

<u>IDMSLBL</u>	Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file.
	►►For a complete listing of IDMSLBL, see Appendix C, “IDMSLBL Procedure for VSE/ESA JCL.”
<u>dmcl-name</u>	The name of a Release 12.0 DMCL
<u>dbname</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>subschema-name</u>	Name of the subschema IDMSRUPD will use if not IDMSRSSD

### VM/ESA commands IDMSRUPD

```
FILEDEF SYSLST PRINTER
FI asfdef DISK cdms asfdef fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
If processing more than one data area, add additional database
file assignments, as required.
FI asfdata DISK cdms asfdata fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dcmsg DISK idms dmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FILEDEF sysjrn1 TAPI SL VOLID nnnnnn (RECFM VB LRECL 111 BLKSIZE bbbb
FILEDEF SYSIDMS DISK sysidms input a
FILEDEF SYSIPT DISK idmsrupd input a
GLOBAL LOADLIB dbalib idmslib
OSRUN IDMSRUPD
```

---

<u>ASFDEF</u>	DDname of the ASF definition area
<u>CDMS ASFDEF FM</u>	File identifier of the ASF definition area
<u>ASFDATA</u>	DDname of the ASF data area
<u>CDMS ASFDATA FM</u>	File identifier of the ASF data area
<u>DCMMSG</u>	DDname of the system message (DDLDCMSG) area
<u>IDMS DMSGDB FM</u>	File identifier of the system message (DDLDCMSG) area
<u>PPP</u>	Page size of the database file
<u>NNN</u>	Number of pages in the database file
<u>SYSJRN1</u>	DDname of the tape journal file, if journaling
<u>NNNNNN</u>	Volume serial number of the tape journal file
<u>LLL</u>	Record length of the tape journal file
<u>BBBB</u>	Block size of the tape journal file
<u>SYSIDMS INPUT A</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ dmcl-name — name of Release 12.0 DMCL</li> <li>■ dbname or segment-name — database name or segment name identifying the DDLDML area of the ASF dictionary</li> </ul>
<u>IDMSRUPD INPUT A</u>	File identifier of the file containing the name of the subschema IDMSRUPD will use if not IDMSRSSD
<u>DBALIB</u>	Filename of the load library containing the DMCL and database name table load modules
<u>IDMSLIB</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules

---

**BS2000/OSD JCL IDMSRUPD**

```

/ FILE    LINK=CDMSLIB,idms.dbalib
/ FILE    LINK=CDMSLIB1,idms.loadlib
/ FILE    LINK=CDMSLDR,idms.loadlib
/ FILE    LINK=CDMSPAM,lodrwork,SPACE=(primlodr,seclodr)
/ FILE    LINK=asfdef,idms.asfdef,SHARUPD=YES
If processing more than one data area, add /FILE statements for them.
/ FILE    LINK=asfdata,idms.asfdata,SHARUPD=YES
/ FILE    LINK=dcmsg,idms.sysmsg.ddldcmsg,SHARUPD=YES
/ FILE    LINK=sysjrn1,idms.tapejrn1
/ FILE    LINK=SYSIDMS,*DUMMY
/ SYSFILE SYSDTA=(SYSCMD)
/ EXEC    (IDMSRUPD,idms.loadlib)

Insert SYSIDMS parameters here

DMCL=dmc1-name
DBNAME=dbname or segment-name
END-SYSIDMS

Insert IDMSRUPD parameters here

SUB=subschema-name

```

<u>idms.dba.loadlib</u>	Filename of the load library containing the DMCL and database name table load modules
<u>idms.loadlib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules
<u>idms.asfdef</u>	Filename of the ASF definition area
<u>idms.asfdata</u>	Filename of the ASF (DDLDML) dictionary area
<u>idms.sysmsg.ddldcmsg</u>	Filename of the system message (DDLDCMSG) area
<u>idms.tapejrn1</u>	Filename of tape journal file
<u>dmc1-name</u>	The name of a Release 12.0 DMCL
<u>dbname or segment-name</u>	Database name or segment name identifying the DDLDML area of the ASF dictionary
<u>subschema-name</u>	Name of the subschema IDMSRUPD will use if not IDMSRSSD

**8.2.9 Output from IDMSRUPD**

**Output from IDMSRUPD:** The IDMSRUPD program produces an output listing identifying the records processed and whether or not the RDEFREC was updated with a db-key.

**Sample IDMSRUPD output listing:** A portion of the output listing produced by the IDMSRUPD program is provided below.

All errors that result from the execution of the IDMSRUPD program are due to a mismatch between the OOAK record and the table definition. When any error results, the RDEF record is *not* updated with a db-key.

Date/time in OOAK record is not the same as found in the RDEF record.

Status of table is not 'G' for generated.

```

STARTING TO UPDATE IDMSR-AREA

PROCESSING RECORD ID: 31998 RFUR-000102-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000102

PROCESSING RECORD ID: 31996 RFUR-000101-00AK
TABLE NOT GENERATED FOR TDN: 000101

PROCESSING RECORD ID: 31994 RFUR-000103-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000103

PROCESSING RECORD ID: 31992 RFUR-000104-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000104

PROCESSING RECORD ID: 31990 RFUR-000106-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000106

PROCESSING RECORD ID: 31988 RFUR-000105-00AK
RECORD: RFUR-000105-00AK
NOT FOUND IN AREA: IDMSR-AREA2

PROCESSING RECORD ID: 31986 RFUR-000108-00AK
DATE/TIME MISMATCH FOR TDN: 000108

PROCESSING RECORD ID: 31984 RFUR-000114-00AK
DATE/TIME MISMATCH FOR TDN: 000114

PROCESSING RECORD ID: 31982 RFUR-000109-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000109

PROCESSING RECORD ID: 31980 RFUR-000111-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000111

PROCESSING RECORD ID: 31974 RFUR-000137-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000137

PROCESSING RECORD ID: 31972 RFUR-000115-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000115

PROCESSING RECORD ID: 31970 RFUR-000116-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000116

PROCESSING RECORD ID: 31968 RFUR-000117-00AK
TABLE NOT GENERATED FOR TDN: 000117

PROCESSING RECORD ID: 31966 RFUR-000118-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000118

PROCESSING RECORD ID: 31964 RFUR-000205-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000205

PROCESSING RECORD ID: 31962 RFUR-000120-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000120

PROCESSING RECORD ID: 31960 RFUR-000121-00AK
DBKEY UPDATED IN RDEFREC FOR TDN: 000121

```

## 8.2.10 Optionally, add CALC record to IDMSR schema

If the IDMSRSSM output listing indicates that you need to add a CALC record to the IDMSR schema, modify the IDMSR schema using the 12.0 schema compiler and the following syntax:

```

MODIFY SCHEMA NAME IS IDMSR VERSION IS 1.
ADD RECORD NAME IS ASF-DUMMY-REC
RECORD ID IS 9037
LOCATION MODE IS CALC USING OWNER-RECORD-DEF-NUMBER
DUPLICATES ARE NOT ALLOWED
WITHIN AREA IDMSR-AREA.
02 OWNER-RECORD-DEF-NUMBER
    PICTURE IS S9(8)
    USAGE IS COMP
    COMMENTS 'DUMMY CALC RECORD FOR IDMSR SCHEMA'.
VALIDATE.
```

## 8.2.11 Execution JCL

Sample execution JCL to execute the batch schema compiler is provided below.

### OS/390 JCL: IDMSCHEM

```

//UPDSTEP EXEC PGM=IDMSCHEM,REGION=
//STEPLIB DD DSN=idms.loadlib,DISP=SHR
//asfdict DD DSN=idms.asfdict.ddldml,DISP=SHR
//dcmsg DD DSN=idms.sysmsg.ddldcmsg,DISP=SHR
//sysjrn1 DD DSN=idms.tapejrn1,DISP=(NEW,KEEP),UNIT=TAPE
//SYSLST DD SYSOUT=A
//SYSRCH DD SYSOUT=A
//SYSIDMS DD *
DMCL=dmcl-name
DICTNAME=dictionary-name
//SYSIPT DD *
Insert modified IDMSR schema statements here
/*

```

<u>idms.loadlib</u>	Data set name of the load library containing executable Release 12.0 CA-IDMS modules
<u>asfdict</u>	DDname of the ASF (DDLDML) dictionary area
<u>idms.asfdict.ddldml</u>	Data set name of the ASF (DDLDML) dictionary area
<u>dcmsg</u>	DDname of the system message (DDLDCMSG) area
<u>idms.sysmsg.ddldcmsg</u>	Data set name of the system message (DDLDCMSG) area
<u>sysjrn1</u>	DDname of the tape journal file, if journaling

---

<u>idms.tapejrl</u>	Dataset name of the tape journal file, if journaling
---------------------	--

---

**VSE/ESA JCL: IDMSCHEM**

```
// JOB IDMSCHEM
// LIBDEF *,SEARCH=120 libraries
// EXEC PROC=idmslbls
// ASSGN sysnnn,DISK,VOL=nnnnnn,SHR
// ASSGN SYSPCH,DISK,VOL=nnnnnn,SHR
// DBLBL IJSYSPH,'syspch.dsn'
// EXTENT SYSPCH,volume,x,x,x,x
// EXEC IDMSCHEM,SIZE=1048K
DMCL=dmc1-name
DBNAME=dbname or segment-name
/*
   Insert modified IDMSR schema statements here
/*
```

**IDMSLBLS**

Name of the procedure provided at installation that contains the file definitions for CA-IDMS dictionaries, databases, disk journal files, and the SYSIDMS file.

►►For a complete listing of IDMSLBLS, see Appendix C, “IDMSLBLS Procedure for VSE/ESA JCL.”

**VM/ESA commands: IDMSCHEM**

```
FILEDEF SYSLST PRINTER
FILEDEF SYSPCH PRINTER
FILEDEF sysjrl TAP1 SL VOLID nnnnnn (RECFM VB LRECL lll BLKSIZE bbbb
FI asfdict DISK cdms asfdict fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnn
FI dcmsg DISK cdms dmsgdb fm (RECFM F LRECL ppp BLKSIZE ppp XTENT nnm
FILEDEF SYSIDMS DISK sysidms input a
FILEDEF SYSIPT DISK schema input a
GLOBAL LOADLIB idmslib
OSRUN IDMSCHEM
```

---

<u>sysjrl</u>	DDname of the tape journal file, if journaling
---------------	--

---

<u>nnnnnn</u>	Volume serial number of the tape journal file
---------------	---

---

<u>lll</u>	Record length of the tape journal file
------------	--

---

<u>bbbb</u>	Block size of the tape journal file
-------------	-------------------------------------

---

<u>asfdict</u>	DDname of the ASF (DDLDML) dictionary area
----------------	--

---

<u>cdms asfdict fm</u>	File identifier of the ASF (DDLDML) dictionary area
------------------------	---

---

<u>dcmsg</u>	DDname of the system message (DDLDCMSG) area
--------------	--

---

---

<u>cdms dmsgdb fm</u>	File identifier of the system message (DDLDCMSG) area
<u>ppp</u>	Page size of the database file
<u>nnn</u>	Number of pages in the database file
<u>sysidms input a</u>	File identifier of the file containing the following: <ul style="list-style-type: none"> <li>■ <i>dmcl-name</i> — name of the DMCL module</li> <li>■ <i>dictionary-name</i> — name of the dictionary containing the schema to be updated</li> </ul>
<u>schema input a</u>	File identifier of the file containing the IDMSR schema statements
<u>idmslib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules

---

**BS2000/OSD JCL: IDMSCHEM**

```

/ FILE      LINK=CDMSLIB,idms.dbalib
/ FILE      LINK=CDMSLIB1,idms.loadlib
/ FILE      LINK=CDMSLDR,idms.loadlib
/ FILE      LINK=CDMSPAM,ldrwork,SPACE=(primladr,secladr)
/ FILE      LINK=asfdict,idms.asfdict.ddldml,SHARUPD=YES
/ FILE      LINK=dcmsg,idms.sysmsg.ddldcmsg,SHARUPD=YES
/ FILE      LINK=sysjrn1,idms.tapejrn1
/ FILE      LINK=SYSIDMS,*DUMMY
/ SYSFILE   SYSDTA=(SYSCMD)
/ EXEC      (IDMSCHEM,idms.loadlib)

```

Insert SYSIDMS parameters here

```

DMCL=dmcl-name
DICTNAME=dictionary-name
END-SYSIDMS

```

Insert modified IDMSR schema statements here

---

<u>idms.loadlib</u>	Filename of the load library containing executable Release 12.0 CA-IDMS modules
<u>idms.asfdict.ddldml</u>	Filename of the ASF (DDLDML) dictionary area
<u>idms.sysmsg.ddldcmsg</u>	Filename of the system message (DDLDCMSG) area
<u>idms.tapejrn1</u>	Filename of the tape journal file, if journaling

---

# Chapter 9. Application Environment

---

9.1 About this chapter . . . . .	9-3
9.2 Local mode processing . . . . .	9-4
9.2.1 SYSIDMS parameter file . . . . .	9-4
9.2.2 Security in local mode . . . . .	9-4
9.3 Deleted CA-IDMS modules . . . . .	9-5
9.4 New version of IDMSUTIO . . . . .	9-6
9.5 COBOL (IDMSDMLC) precompiler JCL . . . . .	9-7
9.6 PL/I (IDMSDMLP) precompiler JCL . . . . .	9-8
9.7 Considerations for relinking application programs . . . . .	9-9
9.8 SPF indexes . . . . .	9-11
9.9 Region size . . . . .	9-12
9.10 CA-ADS . . . . .	9-13



## 9.1 About this chapter

This chapter identifies some of the changes you must make to your CA-IDMS application environment for Release 12.0.

This chapter is organized as follows:

- Local mode processing
- Deleted CA-IDMS module
- New version of IDMSUTIO
- New COBOL and PL/I precompiler JCL
- Considerations for relinking applications
- SPF indexes
- Region size
- CA-ADS

## 9.2 Local mode processing

### 9.2.1 SYSIDMS parameter file

**What is the SYSIDMS parameter file:** Applications that run in a local mode environment will use the SYSIDMS parameter file in JCL streams to specify:

- Physical requirements of the environment such as the DMCL and database to use at run time
- Run-time directives that assist in application execution
- Operating system-dependent file information

►► For a description of SYSIDMS parameters, see either *CA-IDMS Features Summary (Release 12.0)* or *CA-IDMS Database Administration*.

**Review JCL stream for local mode programs:** You should review the JCL streams for all batch programs that run in local mode and:

- Identify the SYSIDMS parameters needed by the program
- Code a SYSIDMS parameter file
- Remove unnecessary parameters from the SYSIPT file
- Relink the program in a 12.0 environment

Note that the DMCL name defaults to IDMSDMCL in local mode. If you will use a different DMCL name, be sure to explicitly code it in the SYSIDMS parameter file.

**Other CA-IDMS products using SYSIDMS:** In addition to user-written batch applications, be sure to review and modify the JCL streams for batch jobs that execute:

- CA-ADS/Batch
- CA-CULPRIT
- CA-OLQ

### 9.2.2 Security in local mode

Based on how 12.0 central security is implemented in your environment, you may need to add additional JCL statements to the execution JCL for programs running in local mode to define the system dictionary and user catalog.

►► For more information on security in local mode, see *CA-IDMS Security Administration*.

## 9.3 Deleted CA-IDMS modules

**Remove references to deleted modules:** If any of your 10.2 applications are linked with the CA modules listed below, remove the reference and link edit the program in a 12.0 environment.

- IDMSINTB
- IDMSQSAM
- IDMSTRAC

**SYSIDMS replaces the need for IDMSQSAM and IDMSTRAC:** SYSIDMS parameters replace the need for IDMSQSAM and IDMSTRAC.

The only way to use IDMSQSAM is to specify it in a SYSIDMS parameter file.

Create a SYSIDMS parameter file in the execution JCL of 10.2 programs that link to IDMSQSAM and IDMSTRAC and link edit the program in a 12.0 environment.

## 9.4 New version of IDMSUTIO

**New 12.0 version of IDMSUTIO:** If any of your batch programs are linked with IDMSUTIO to make use of file I/O routines, you must relink the program with the 12.0 version of IDMSUTIO.

If you use IDMSUTIO for DBNAME and DBNODE processing, relink the program without IDMSUTIO and remove the parameters from the SYSIPT file. Specify DBNAME and DBNODE parameters using the SYSIDMS parameter file.

## 9.5 COBOL (IDMSDMLC) precompiler JCL

There is new CA-IDMS COBOL precompiler (IDMSDMLC) JCL. Review the execution JCL for CA-IDMS COBOL programs that use IDMSDMLC and make the necessary modifications.

- For a description of IDMSDMLC JCL, see *CA-IDMS DML Reference - COBOL*.

## 9.6 PL/I (IDMSDMLP) precompiler JCL

There is new CA-IDMS PL/I precompiler (IDMSDMLP) JCL. Review the execution JCL for CA-IDMS PL/I programs that use IDMSDMLP and make the necessary modifications.

- For a description of IDMSDMLP JCL, see *CA-IDMS DML Reference - PL/I*.

## 9.7 Considerations for relinking application programs

**Changes only necessary on the next recompile or relink:** Changes are necessary in the link-edit parameters of some CA-IDMS application programs. In all cases, except DC-BATCH applications, these changes are necessary *only when you next recompile or relink a program*; until then, existing load modules will execute successfully.

**IDMS is the only entry module:** These changes are necessary, because in Release 12.0, IDMS is the one and only entry module to CA-IDMS services in batch and DC/UCF execution environments.

IDMS contains several entry points to distinguish different types of services (such as navigational or SQL database requests) or different host languages (such as COBOL or PL/I) in DC/UCF.

Upward compatibility is provided for application programs linked with 10.0 interface modules (IDMSIDVS, IDMSCOBI, IDMSPLI, etc.), but these modules are not distributed as part of the 12.0 software and must be replaced with IDMS the next time the programs are linked.

**Considerations by application protocol:** Here are considerations by application protocol:

- IDMS-DC COBOL programs

The next time it is necessary to relink a COBOL program that has a protocol mode of IDMS-DC, you must include module IDMS instead of IDMSCOBI with your application program.

- IDMS-DC PL/I programs

The next time it is necessary to recompile a PL/I program that has a protocol mode of IDMS-DC, you must do the following:

Add the following declaration to your program source:

DECLARE IDMSPLI ENTRY OPTIONS (INTER, ASSEMBLER)

Link it with the module IDMS instead of IDMSPLI

- DC-BATCH programs

All programs, regardless of language, which have a protocol mode of DC-BATCH, must be relinked before being executed in a 12.0 environment. In the new link-edit, replace both IDMSDCBI and IDMSINTB with IDMS.

PL/I programs must also be recompiled before being executed. You must add the following declaration to your program source:

DECLARE IDMSDCP ENTRY OPTIONS (INTER, ASSEMBLER)

- Batch programs in a VSE/ESA environment

The next time it is necessary to relink a COBOL or PL/I program that has a protocol mode of BATCH, you must include the IDMS module and remove IDMSIDVS or IDMSLDPT.

In Release 12.0, the CA-IDMS batch environment always uses GETVIS for storage acquisition. When executing programs in a 12.0 environment which were linked with 10.0 IDMS and IDMSLDPT (using COMREG for storage management), it may be necessary to increase the size of the partition or relink the application using the 12.0 IDMS.

## 9.8 SPF indexes

You must convert SPF indexes to integrated indexes before migrating applications that use SPF indexes to a 12.0 environment.

Modify 10.2 schema definitions containing SPF indexes and create appropriate 10.2 subschemas that replace SPF indexes with integrated indexes. Then execute the 10.2 IDMSTBLU utility program to convert SPF indexes to integrated indexes.

After converting SPF indexes in a 10.0 environment, you may want to use the 12.0 UNLINKED index option. If so, you will need to restructure the database in the 12.0 environment. For instructions on restructuring a database, refer to *CA-IDMS Utilities*.

If an application contains BIND statements for SPF system records, remove the BIND statements and recompile the program.

## 9.9 Region size

You may need to increase the region size for local mode applications to accommodate the increase in the size of the enhanced local mode environment.

## 9.10 CA-ADS

**Upward compatibility:** All of your CA-ADS Release 10.21 applications will run, without program modifications, in a Release 12.0 environment.

## 9.10 CA-ADS

---

## **Appendix A. Dictionary Segments**

---

A.1 About this appendix . . . . .	A-3
A.2 System and application dictionary segments . . . . .	A-4



## A.1 About this appendix

This appendix includes a list of the segments, files, and areas that comprise the dictionaries and databases provided at installation.

## A.2 System and application dictionary segments

Here is a list of the segments that comprise the installed system and application dictionaries.

<b>Segment</b>	<b>File</b>	<b>Assign</b>	<b>Area</b>
SYSTEM	SYSTEM.DCDML	DCDML	SYSTEM.DDLDDML
	SYSTEM.DCRUN	DCRUN	SYSTEM.DDLDCRUN
	SYSTEM.DCLOG	DCLOG	SYSTEM.DDLDCLOG
	SYSTEM.DCSCR	DCSCR	SYSTEM.DDLDCSCR
	SYSTEM.DCLOD	DCLOD	SYSTEM.DDLDCLOD
CATSYS	CATSYS.DCCAT	DCCAT	CATSYS.DDLCAT
	CATSYS.DCCATX	DCCATX	CATSYS.DDLCATX
	CATSYS.DCCATL	DCCAT	CATSYS.DDLCATLOD
APPLDICT	APPLDICT.DICTDB	DICTDB	APPLDICT.DDLDDML
	APPLDICT.DLODDB	DLODDB	APPLDICT.DDLDCLOD
SYSDIRL	SYSDIRL.DIRLDB	DIRLDB	SYSDIRL.DDLDDML
	SYSDIRL.DIRLLOD	DIRLLOD	SYSDIRL.DDLDCLOD
SYSLOC	SYSLOC.DCLOC	DCLOC	SYSLOC.DDLOCSCR
SYSMSG	SYSMSG.DCMMSG	DCMSG	SYSMSG.DDLDCMSG
SYSUSER	SYSUSER.SECDD	SECDD	SYSUSER.DDLSEC
SYSSQL	SYSSQL.SQLDD	SQLDD	SYSSQL.DDLCAT
	SYSSQL.SQLXDD	SQLXDD	SYSSQL.DDLCATX
	SYSSQL.SQLLOD	SQLLOD	SYSSQL.DDLCATLOD
ASFdict	ASFdict.ASFDM	ASFDM	ASFdict.DDLDDML
	ASFdictASFLOD	ASFLOD	ASFdict.DDLDCLOD
	ASFdict.DEFN	DEFN	ASFdict.IDMSR-AREA
	AFSDict.DATA	DATA	AFSDict.IDMSR-AREA2
EMPDEMO	EMPDEMO.EMPDEMO	EMPDEMO	EMPDEMO.EMPDEMO
	EMPDEMO.INSDEMO	INSDEMO	EMPDEMO.INSDEMO
	EMPDEMO.ORGDEMO	ORGDEMO	EMPDEMO.ORGDEMO
SQLDEMO	SQLDEMO.EMPLDEMO	EMPLDEMO	SQLDEMO.EMPLAREA
	SQLDEMO.INFODEMO	INFODEMO	SQLDEMO.INFOAREA

Segment	File	Assign	Area
	SQLDEMO.INDXDEMO	INDXDEMO	SQLDEMO.INDXAREA
PROJSEG	PROJSEG.PROJDEMO	PROJDEMO	PROJSEG.PROJAREA

- SYSDIRL contains IDMSDIRL output plus CULPRIT report source.
- ASFdict is optional. You only need it if you are installing ASF.
- SYSSQL is optional. You only need it if you are installing the CA-IDMS/SQL Option.
- SQLDEMO is an optional SQL-defined sample database for use with the SQL Option. It is defined in SYSSQL.
- EMPDEMO is an optional non SQL-defined sample database. It is defined in APPLDICT.
- SYSLOC is a scratch area used in local mode. It is optional. If it is not present, DDLDCSCR is used instead.



## **Appendix B. R120DMCL and R120DBTB Listing**

---

B.1 About this appendix . . . . .	B-3
B.2 R120DMCL listing . . . . .	B-4
B.3 R120DBTB database name table . . . . .	B-6



## B.1 About this appendix

This appendix provides the IDMSLOOK reports for the R120DMCL DMCL and the R120DBTB database name table.

## B.2 R120DMCL listing

This is an IDMSLOOK report of the Release 12.0 DMCL that defines the 12.0 environment provided at installation.

This DMCL uses dbtable R120DBT8				The Operating System is OS				
Area Name	Page Group	Low Page	High Page	Page Size	DDNAME	File Name	Low Page	High Page
*****	*****	*****	*****	*****	*****	*****	*****	*****
APPLDICT.DDLML	0	60,001	62,000	4,276	DICTDB	APPLDICT.DICTDB	*** SAME ***	*** SAME ***
APPLDICT.DDLDCLOD	0	70,001	70,500	4,276	DLODDB	APPLDICT.DLODDB	*** SAME ***	*** SAME ***
ASFIDICT.DDLML	0	80,001	82,000	4,276	ASFML	ASFIDICT.ASFML	*** SAME ***	*** SAME ***
ASFIDICT.IDMSR-AREA	0	83,001	84,000	4,276	ASFDEFN	ASFIDICT.ASFDEFN	*** SAME ***	*** SAME ***
ASFIDICT.IDMSR-AREA2	0	85,001	87,000	4,276	ASFDATA	ASFIDICT.ASFDATA	*** SAME ***	*** SAME ***
ASFIDICT.DDLDCLOD	0	88,001	90,000	4,276	ASFLOD	ASFIDICT.ASFLOD	*** SAME ***	*** SAME ***
CATSYS.DDLCAT	0	1	300	4,276	DCCAT	CATSYS.DCCAT	*** SAME ***	*** SAME ***
CATSYS.DDLCATX	0	801	900	4,276	DCCATX	CATSYS.DCCATX	*** SAME ***	*** SAME ***
CATSYS.DDLCATL	0	901	950	4,276	DCCATL	CATSYS.DCCATL	*** SAME ***	*** SAME ***
EMPDEMO.EMP-DEMO-REGION	0	75,001	75,050	4,276	EMPDEMO	EMPDEMO.EMPDEMO	*** SAME ***	*** SAME ***
EMPDEMO.INS-DEMO-REGION	0	75,101	75,125	4,276	INSDEMO	EMPDEMO.INSDEMO	*** SAME ***	*** SAME ***
EMPDEMO.ORG-DEMO-REGION	0	75,151	75,175	4,276	ORGDEMO	EMPDEMO.ORGDEMO	*** SAME ***	*** SAME ***
PROJSEG.PROJAREA	0	77,401	77,450	4,276	PROJDEMO	PROJSEG.PROJDEMO	*** SAME ***	*** SAME ***
SQLDEMO.EMPLAREA	0	77,001	77,100	4,276	EMPLDEMO	SQLDEMO.EMPLDEMO	*** SAME ***	*** SAME ***
SQLDEMO.INFOAREA	0	77,201	77,250	4,276	INFODEMO	SQLDEMO.INFODEMO	*** SAME ***	*** SAME ***
SQLDEMO.INDXAREA	0	77,301	77,350	4,276	INDXDEMO	SQLDEMO.INDXDEMO	*** SAME ***	*** SAME ***
SYSDIRL.DDLDCLOD	0	4,001	4,010	4,276	DIRLLOD	SYSDIRL.DIRLLOD	*** SAME ***	*** SAME ***
SYSDIRL.DDLML	0	5,001	7,000	4,276	DIRLDB	SYSDIRL.DIRLDB	*** SAME ***	*** SAME ***
SYSLOC.DDLLOCSCR	0	55,001	57,000	4,276	DCLSCR	SYSLOC.DCLSCR	*** SAME ***	*** SAME ***
SYSMSG.DDLDCMSG	0	10,001	14,000	4,276	DCMSG	SYSMSG.DCMSG	*** SAME ***	*** SAME ***
SYSSQL.DDLCAT	0	20,001	22,000	4,276	SQLDD	SYSSQL.SQLDD	*** SAME ***	*** SAME ***
SYSSQL.DDLCATL	0	25,001	25,500	4,276	SQLLOD	SYSSQL.SQLLOD	*** SAME ***	*** SAME ***
SYSSQL.DDLCATX	0	28,001	28,500	4,276	SQLXDD	SYSSQL.SQLXDD	*** SAME ***	*** SAME ***
SYSTEM.DDLML	0	1,001	2,000	4,276	DCDML	SYSTEM.DCDML	*** SAME ***	*** SAME ***
SYSTEM.DDLDCLOD	0	3,001	3,100	4,276	DCLOD	SYSTEM.DCLOD	*** SAME ***	*** SAME ***
SYSTEM.DDLDCLOG	0	30,001	34,000	4,276	DCLOG	SYSTEM.DCLOG	*** SAME ***	*** SAME ***
SYSTEM.DDLDCRUN	0	40,001	41,000	2,676	DCRUN	SYSTEM.DCRUN	*** SAME ***	*** SAME ***
SYSTEM.DDLDCSCR	0	50,001	52,000	2,676	DCSCR	SYSTEM.DCSCR	*** SAME ***	*** SAME ***
SYSUSER.DDLSEC	0	48,001	48,500	4,276	SECDD	SYSUSER.SECDD	*** SAME ***	*** SAME ***

File Name	DDNAME	Type	Buffer Name	Data Set Name (DSN)
*****	*****	****	*****	*****
APPLDICT.DICTDB	DICTDB	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.APPLDICT.DDLML
APPLDICT.DLODDB	DLODDB	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.APPLDICT.DDLCLOD
ASFDCICT.ASFDATA	ASFDATA	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.ASFDCICT.ASFDATA
ASFDCICT.ASFDEFN	ASFDEFN	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.ASFDCICT.ASFDEFN
ASFDCICT.ASFDML	ASFML	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.ASFDCICT.DDLML
ASFDCICT.ASFLOD	ASFLOD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.ASFDCICT.ASFLOD
CATSYS.DCCAT	DCCAT	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.CATSYS.DCCAT
CATSYS.DCCATL	DCCATL	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.CATSYS.DCCATL
CATSYS.DCCATX	DCCATX	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.CATSYS.DCCATX
EMPDEMO.EMPDEMO	EMPDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.EMPDEMO.EMPDEMO
EMPDEMO.INSDEMO	INSDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.EMPDEMO.INSDEMO
EMPDEMO.ORGDEMO	ORGDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.EMPDEMO.ORGDEMO
PROJSEG.PROJDEMO	PROJDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.PROJSEG.PROJDEMO
SQLDEMO.EMPLDEMO	EMPLDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SQLDEMO.EMPLDEMO
SQLDEMO.INDXDEMO	INDXDEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SQLDEMO.INDXDEMO
SQLDEMO.INFODEMO	INFODEMO	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SQLDEMO.INFODEMO
SYSDIRL.DIRLDB	DIRLDB	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSDIRL.DDLML
SYSDIRL.DIRLLOD	DIRLLOD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSDIRL.DDLCLOD
SYSLOC.DCLSCR	DCLSCR	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSLOC.DDLCLOD
SYMSG.DCMSG	DCMSG	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYMSG.DDLCMSG
SYSSQL.SQLDD	SQLDD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSSQL.DDLCAT
SYSSQL.SQLLDD	SQLLDD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSSQL.DDLCATL
SYSSQL.SQLXDD	SQLXDD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSSQL.DDLCATX
SYSTEM.DCDML	DCDML	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSTEM.DDLML
SYSTEM.DCLOD	DCLOD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSTEM.DDLCLOD
SYSTEM.DCLOG	DCLOG	BDAM	LOG_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSTEM.DDLCLLOG
SYSTEM.DCRUN	DCRUN	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSTEM.DDLDCRUN
SYSTEM.DCSRC	DCSRC	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSTEM.DDLDCSCR
SYSUSER.SECDD	SECDD	BDAM	DEFAULT_BUFFER	DISP=SHR,DSN=DBDC.SYSTEM82.SYSUSER.DDLSEC

DMCL Journals	Page Size	# of Pages						
*****	*****	*****						
J1JRNL	2,004	5,000						
J2JRNL	2,004	5,000						
J3JRNL	2,004	5,000						
J4JRNL	2,004	5,000						
SYSJRNL	8,000	Archive						
Journal Buffers	Buffer Size	# of Buffers						
*****	*****	*****						
JNL_BUFFER	2,004	5						
DMCL Buffers	Buffer Size	CV Buffers	CV Type	Total CV Size	Local Buffers	Local Type	Total Local Size	
*****	*****	*****	**	*****	*****	**	*****	
LOG_BUFFER	4,276	5	OS	21,380	5	DC	21,380	
DEFAULT_BUFFER	4,276	30	OS	128,280	20	OS	85,520	
0 Bytes used for CV buffers in DC storage			149,660 Bytes used for CV buffers in OS storage			149,660 Bytes used for CV DMCL Buffers		
21,380 Bytes used for LOCAL buffers in DC storage			85,520 Bytes used for LOCAL buffers in OS storage			106,900 Bytes used for LOCAL DMCL Buffers		

## B.3 R120DBTB database name table

Here is an IDMSLOOK utility report on the R120DBTB database name table. This is the database name table provided at installation that describes the databases accessible by the R120DMCL.

```
Dbtable=R120DBTB          Compiled Date=91-12-10 02.30.12
The DEFAULT Dictionary is SYSDICT

DBNAME is *DEFAULT      match on subschema is OPTIONAL
  Subschema IDMSNWK? maps to IDMSNWK? using DBNAME → SYSDICT
  Subschema IDMSCAT? maps to IDMSCAT? using DBNAME → SYSDICT

DBNAME is ASFDICT      match on subschema is OPTIONAL
  Include SEGMENT → ASFDICT
  Include SEGMENT → SYSMSG

DBNAME is SYSDICT      match on subschema is OPTIONAL
  Include SEGMENT → APPLDICT
  Include SEGMENT → SYSMSG
  Include SEGMENT → SYSSQL

DBNAME is SYSDIRL      match on subschema is OPTIONAL
  Include SEGMENT → SYSDIRL
  Include SEGMENT → SYSMSG

DBNAME is SYSTEM        match on subschema is OPTIONAL
  Include SEGMENT → CATSYS
  Include SEGMENT → SYSMSG
  Include SEGMENT → SYSTEM

IDMSLOOK - 1 Selection Card Processed
```

## **Appendix C. IDMSLBLS Procedure for VSE/ESA JCL**

---

C.1 About this appendix . . . . .	C-3
C.2 IDMSLBLS Procedure . . . . .	C-4



## C.1 About this appendix

This appendix lists the IDMSLBLS procedure referenced in VSE/ESA JCL in this document.

## C.2 IDMSLBLS Procedure

**What is the IDMSLBLS procedure:** IDMSLBLS is a procedure provided during a CA-IDMS VSE/ESA installation. It contains file definitions for the CA-IDMS components listed below. These components are provided during installation:

- Dictionaries
- Sample databases
- Disk journal files
- SYSIDMS file

Tailor the IDMSLBLS procedure to reflect the filenames and definitions in use at your site and include this procedure in VSE/ESA JCL job streams.

The sample VSE/ESA JCL provided in this document includes the IDMSLBLS procedure. Therefore, individual file definitions for CA-IDMS dictionaries, sample databases, disk journal files, and SYSIDMS file are not included in the sample JCL.

### **IDMSLBLS procedure listing**

```

* ----- LIBDEFS -----
// LIBDEF *,SEARCH=idmslib.sublib
// LIBDEF *,CATALOG=user.sublib
/* ----- LABELS -----
// DLBL idmslib,'idms.library',2099/365
// EXTENT ,nnnnnn,,ssss,1500
// DLBL dccat,'idms.system.dccat',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,31
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dccat1,'idms.system.dccat1od',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dccatx,'idms.system.dccatx',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,11
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dcndl,'idms.system.ddldndl',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,101
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dcldod,'idms.system.ddldcldod',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,21
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dclog,'idms.system.ddldclog',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,401
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dcrun,'idms.system.ddldcrun',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,68
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dcscr,'idms.system.ddldcscr',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,135
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dcmsg,'idms.sysmsg.ddldcmgsq',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,201
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dclscr,'idms.sysloc.ddlocscr',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dirldb,'idms.sysdirl.ddldml',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,201
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dirldod,'idms.sysdirl.ddldcldod',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,2
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL empdemo,'idms.empdemo1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,11
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL insdemo,'idms.insdemo1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL orgdemo,'idms.orgdemo1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL empldem,'idms.sqldemo.emp1demo',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,11
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL infodem,'idms.sqldemo.infodemo',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL projdem,'idms.projseg.projdemo',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6

```

```

// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL idxdem,'idms.sqldemo.indxdemo',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,6
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL sysct1,'idms.sysct1',2099/365,SD
// EXTENT SYSnnn,nnnnnn,,ssss,2
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL secdd,'idms.sysuser.ddlsec',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,26
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dictdb,'idms.appldict.ddldml',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,51
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL dloddb,'idms.appldict.ddldclod',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,51
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL sgddd,'idms.syssql.ddlcat',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,101
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL sg1lod,'idms.syssql.ddlcat1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,51
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL sg1xdd,'idms.syssql.ddlcatx',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,26
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL asfdml,'idms.asfdict.ddldml',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,201
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL asflod,'idms.asfdict.asflod',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,401
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL asfdata,'idms.asfdict.asfdata',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,201
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL ASFDEFN,'idms.asfdict.asfdefn',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,101
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL j1jrn1,'idms.j1jrn1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,54
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL j2jrn1,'idms.j2jrn1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,54
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL j3jrn1,'idms.j3jrn1',2099/365,DA
// EXTENT SYSnnn,nnnnnn,,ssss,54
// ASSGN SYSnnn,DISK,VOL=nnnnnn,SHR
// DLBL SYSIDMS,'#SYSIPT',0,SD
/*
*/

```

<u>idmslib.sublib</u>	Name of the sublibrary within the library containing CA-IDMS modules
<u>user.sublib</u>	Name of the sublibrary within the library containing user modules
<u>idmslib</u>	Filename of the file containing CA-IDMS modules
<u>idms.library</u>	File-ID associated with the file containing CA-IDMS modules
<u>SYSn</u>	Logical unit of the volume for which the extent is effective

<u>nnnnnn</u>	Volume serial identifier of appropriate disk volume
<u>ssss</u>	Starting track (CKD) or block (FBA) of disk extent
<u>dccat</u>	Filename of the system dictionary catalog (DDLCAT) area
<u>idms.system.dccat</u>	File-ID of the system dictionary catalog (DDLCAT) area
<u>dccatl</u>	Filename of the system dictionary catalog load (DDLCATLOD) area
<u>idms.system.dccatlod</u>	File-ID of the system dictionary catalog load (DDLCATLOD) area
<u>dccatx</u>	Filename of the system dictionary catalog index (DDLCATX) area
<u>idms.system.dccatx</u>	File-ID of the system dictionary catalog index (DDLCATX) area
<u>dcdml</u>	Filename of the system dictionary definition (DDLDML) area
<u>idms.system.ddldm1</u>	File-ID of the system dictionary definition (DDLDML) area
<u>dclog</u>	Filename of the system log area (DDLDCLOG) area
<u>idms.system.ddldclog</u>	File-ID of the system log (DDLDCLOG) area
<u>dcrun</u>	Filename of the system queue (DDLDCRUN) area
<u>idms.system.ddldcrun</u>	File-ID of the system queue (DDLDCRUN) area
<u>dcsqr</u>	Filename of the system scratch (DDLDCSCR) area
<u>idms.system.ddldcscr</u>	File-ID of the system scratch (DDLDCSCR) area
<u>dcmsg</u>	Filename of the system message (DDLDCMSG) area
<u>idms.sysmsg.ddldcmgs</u>	File-ID of the system message (DDLDCMSG) area
<u>dclscr</u>	Filename of the local mode system scratch (DDLOCSCR) area
<u>idms.sysloc.ddlocscr</u>	File-ID of the local mode system scratch (DDLOCSCR) area
<u>dirldb</u>	Filename of the IDMSDIRL definition (DDLDML) area
<u>idms.sysdirl.ddldm1</u>	File-ID of the IDMSDIRL definition (DDLDML) area

<u>dirllod</u>	Filename of the IDMSDIRL definition load (DDLDCLOD) area
<u>idms.sysdirl.dirllod</u>	File-ID of the IDMSDIRL definition load (DDLDCLOD) area
<u>empdemo</u>	Filename of the EMPDEMO area
<u>idms.empdemo1</u>	File-ID of the EMPDEMO area
<u>insdemo</u>	Filename of the INSDEMO area
<u>idms.insdemo1</u>	File-ID of the INSDEMO area
<u>orgdemo</u>	Filename of the ORGDEMO area
<u>idms.orgdemo1</u>	File-ID of the ORDDEMO area
<u>empldem</u>	Filename of the EMPLDEMO area
<u>idms.sqldemo.empldemo</u>	File-ID of the EMPLDEMO area
<u>infodem</u>	Filename of the INFODEMO area
<u>idms.sqldemo.infodem</u>	File-ID of the INFODEMO area
<u>projdem</u>	Filename of the PROJDEMO area
<u>idms.projseg.projdemo</u>	File-ID of the PROJDEMO area
<u>indxdem</u>	Filename of the INDXDEMO area
<u>idms.sqldemo.indxdemo</u>	File-ID of the INDXDEMO area
<u>sysctl</u>	Filename of the SYSCTL file
<u>idms.sysctl</u>	File-ID of the SYSCTL file
<u>secdd</u>	Filename of the system user catalog (DDLSEC) area
<u>idms.sysuser.ddlsec</u>	File-ID of the system user catalog (DDLSEC) area
<u>dictdb</u>	Filename of the application dictionary definition area
<u>idms.appldict.ddldml</u>	File-ID of the application dictionary definition (DDLDML) area
<u>dloddb</u>	Filename of the application dictionary definition load area
<u>idms.appldict.ddldclod</u>	File-ID of the application dictionary definition load (DDLDCLOD) area
<u>sqldd</u>	Filename of the SQL catalog (DDLCAT) area
<u>idms.syssql.ddlcat</u>	File-ID of the SQL catalog (DDLCAT) area
<u>sqllod</u>	Filename of the SQL catalog load (DDLCATL) area
<u>idms.syssql.ddlcatl</u>	File-ID of SQL catalog load (DDLCATL) area
<u>sqlxdd</u>	Filename of the SQL catalog index (DDLCATX) area

---

<u>idms.syssql.ddlcatx</u>	File-ID of the SQL catalog index (DDLCATX) area
<u>asfdml</u>	Filename of the asf dictionary definition (DDLDML) area
<u>idms.asfdict.ddldml</u>	File-ID of the asf dictionary definition (DDLDML) area
<u>asflod</u>	Filename of the asf dictionary definition load (ASFLOD) area
<u>idms.asfdict.asflod</u>	File-ID of the asf dictionary definition load (ASFLOD) area
<u>asfdata</u>	Filename of the asf data (ASFDATA) area
<u>idms.asfdict.asfdata</u>	File-ID of the asf data area (ASFDATA) area
<u>ASFDEFN</u>	Filename of the asf data definition (ASFDEFN) area
<u>idms.asfdict.asfdefn</u>	File-ID of the asf data definition area (ASFDEFN) area
<u>j1jrnl</u>	Filename of the first disk journal file
<u>idms.j1jrnl</u>	File-ID of the first disk journal file
<u>j2jrnl</u>	Filename of the second disk journal file
<u>idms.j2jrnl</u>	File-ID of the second disk journal file
<u>j3jrnl</u>	Filename of the third disk journal file
<u>idms.j3jrnl</u>	File-ID of the third disk journal file
<u>SYSIDMS</u>	Filename of the SYSIDMS parameter file

---



## Appendix D. External Security Managers

---

D.1	About this appendix	D-3
D.2	Global considerations	D-4
D.2.1	User exits	D-4
D.2.2	Signon processing	D-4
D.2.3	Resource access validation	D-7
D.3	CA-ACF2 conversion	D-9
D.3.1	Administration	D-9
D.3.2	Subschema security	D-11
D.3.3	Area security	D-12
D.3.4	CA-ACF2 CA-IDMS interface	D-13
D.4	CA-Top Secret conversion	D-15
D.4.1	Administration	D-15
D.4.2	Task security	D-16
D.4.3	Subschema security	D-16
D.4.4	Area security	D-18
D.5	Sample user exit 14 processing	D-19



## D.1 About this appendix

This appendix explains how the central security system in CA-IDMS 12.0 interfaces with CA-ACF2 and CA-Top Secret. It explains how the upgrade to CA-IDMS 12.0 affects the relationship between the external security manager and CA-IDMS.

## D.2 Global considerations

**About this section:** The information in this section applies to how CA-IDMS central security works with any external security manager. Information specific to a particular external security manager is noted as appropriate.

In release 10.2, CA-ACF2 and CA-Top Secret security for CA-IDMS is provided using the CA-ACF2 IDMS and CA-Top Secret security subsystems. You must install these subsystems to use either CA-ACF2 or CA-Top Secret security. CA-ACF2 IDMS and CA-Top Secret IDMS allow ACF2 and Top Secret to be used as an external security manager for CA-IDMS resources.

CA-IDMS 12.0 provides an integrated and centralized security manager so that separate security interfaces such as CA-ACF2 IDMS and CA-Top Secret IDMS do not have to be installed to use CA-ACF2 or CA-Top Secret.

### D.2.1 User exits

Various security exits driven in earlier releases of CA-IDMS have been replaced by exits 28 and 29 in CA-IDMS 12.0. Exits 28 and 29 are driven for SIGNON, SIGNOFF, resource checks, and initialization.

►► For more information on these exits, refer to *CA-IDMS Security Administration*.

### D.2.2 Signon processing

**Password control:** Under CA-IDMS 12.0, one-line signons can be controlled by an optional PTF to RHDCSNON.

**CA-ACF2:** One-line SIGNONS to CA-IDMS Release 10.2 are controlled by QLOGON parameter of @MOPTS in CA-ACF2.

When a user exceeds the password violation threshold defined to the external security manager, the action is not reported to CA-IDMS/DC.

**Password reverification:** In releases of CA-IDMS 12.0 prior to the 9212 maintenance release, password reverification is not performed automatically and is not supported by CA-IDMS components. A return code 16 (X'10') from a #SECHECK indicates the need to reverify the user's password. Beginning with the 9212 maintenance release, password reverification is performed by CA-IDMS central security.

**Automatic terminal signon:** Automatic terminal signon is optional for CA-IDMS 12.0. It can be specified with the DFLTSGN parameter of #SECRTT.

**Note:** DFLTSGN is valid on the 9212 and later maintenance releases of CA-IDMS 12.0.

In processing automatic terminal signon for online tasks, CA-IDMS 12.0 uses:

- The VTAM node name
- The PTERM name, for non-VTAM terminals
- The LTERM name, if there is no physical terminal (for example, with queue-initiated tasks)

For batch jobs under the central version, the logonid of the batch address space (that is, the submitter of the batch job) is used for access validation.

**CA-ACF2:** Under CA-IDMS 10.2, the DFLTLID parameter on the @MOPT macro in CA-ACF2 specified a default logonid used for access validation if no user is signed on.

**CA-Top Secret:** Under CA-IDMS 10.2, the ATS Acidname specifies a default logonid used for access validation if no user is signed on.

**User session attributes:** In CA-IDMS 12.0 attributes for a user session are specified in a system profile and/or a user profile. Certain attribute keywords have meaning to CA-IDMS, such as PRIORITY (user priority) and CLIST (signon CLIST).

►► For more information about attributes and profiles, refer to:

- *CA-IDMS Security Administration*
- *CA-IDMS System Operations*

If the user is defined to CA-IDMS, the system and user profile may be explicitly associated with the user definition.

If the user is not defined to CA-IDMS (user validation is performed externally), you can define a default user profile in the CA-IDMS user catalog, and you can define a default system profile in the system dictionary. The default profiles will be processed at signon if no profiles are associated explicitly with the user definition and:

- The user profile name matches the logonid or the default user profile name specified in the SRTT through the USERPROF parameter of the #SECRTT macro
- The system profile name is 'DEFAULT' or the default system profile name specified in the SRTT through the SYSPROF parameter of the #SECRTT macro

**Note:** The USERPROF and SYSPROF parameters of the #SECRTT macro are valid on the 9212 and subsequent maintenance releases of CA-IDMS.

►► For more information about the SRTT and the #SECRTT macro, refer to *CA-IDMS Security Administration*.

**Multiple signons:** Multiple signons to CA-IDMS are controlled by the MULTIPLE SIGNON clause of the SYSTEM statement in CA-IDMS/DC system generation. This control applies only to signons originating from interactive terminals.

►► For more information about the SYSTEM statement, refer to *CA-IDMS System Generation*.

**CA-ACF2:** Prior to CA-IDMS 12.0, multiple signon was controlled by SIGNQNM.

**CA-Top Secret:** Prior to CA-IDMS 12.0, multiple signon was only controlled by the Top Secret facility option SIGN(m). In CA-IDMS 12.0, you must still specify SIGN(m) along with the CA-IDMS MULTIPLE SIGNON clause to allow multiple signons.

### Other considerations

- **Signon authorization (access to system)**

At signon, a resource check is issued using specifications from the #SECRTT entry for RESTYPE SGON.

**CA-ACF2:** AUTHFLD in the logonid record is not used in CA-IDMS 12.0.

The base resource name matches the specification for SYSTEM ID in the CA-IDMS/DC system generation SYSTEM statement.

►► For more information about the SYSTEM statement, refer to *CA-IDMS System Generation*.

Resource class must be specified in the EXTCLS parameter of the #SECRTT entry. The resource name passed by CA-IDMS central security will be built according to the specifications in the EXTNAME parameter of the #SECRTT macro.

The user must have access to the resource for successful signon to the DC system.

- **Signon message**

CA-IDMS/DC includes a facility to display a "good morning" message at signon. No definition in the external system is required.

- **Security class codes**

Exit 29 can be used to move a bit map of security class codes for the DEFAULT application to the SON after external SIGNON and before internal SIGNON. These security classes will be used if the ACTI resource type has been secured. The DEFAULT application is a mechanism that allows execution of existing applications that use Release 10.2 security classes without having to explicitly define activities for each application.

►► For more information about the ACTI resource type and the DEFAULT application, refer to *CA-IDMS Security Administration*.

### D.2.3 Resource access validation

**Protection by default:** For all resource types in the SRTT supplied with CA-IDMS 12.0, security is turned *off*.

If the entry for a resource type is modified to specify SECBY=INT or SECBY=EXT on the #SECRTT macro, security for all occurrences of that resource type is turned on.

If the resource is not defined in the SRTT, access to the resource is denied.

**Safe lists:** Occurrence overrides of the SRTT security option for a resource type provide a capability equivalent to safe lists.

**CA-ACF2:** Safe list capability for CA-IDMS 10.2 was provided through @GRCE SAFE.

**CA-Top Secret:** Safe list capability for CA-IDMS 10.2 tasks was provided through optional fixes. Safe list capability in 12.0 is provided by occurrence overrides.

CA-IDMS 12.0 supports occurrence overrides for:

- Programs (resource SPGM)
- Tasks (resource type TASK)
- Databases (resource type DB)
- User-defined resource types

If SECBY=OFF is coded on an occurrence override, access is always allowed regardless of SIGNON status; that is, such resources are unsecured for public access and can be accessed by a user who is not signed on.

In this example, the #SECRTT macro specifies an occurrence override that allows public access to program RHDCBYE:

```
#SECRTT RESTYPE=SPGM,
    TYPE=OCCURRENCE,
    RESNAME=RHDCBYE,
    SECBY=OFF
```

**Suspending user IDs for violation threshold:** As distinct from exceeding the password violation threshold, if a user exceeds the violation threshold defined in the external security manager for access to secured resources, this condition is reported by a return code 16 on #SECHECK to CA-IDMS. Given this condition, CA-IDMS ends the session by signing the user off.

**CA-Top Secret:** Violation threshold is defined with the VTHRESH option.

Suspension of the user ID is controlled by the external security manager.

**Controlling where validation will occur:** #SECRTT SECBY=EXTERNAL directs that access to the resource is controlled by the external security manager. #SECRTT SECBY=INT or OFF means that access control, if any, will not be provided by the external security manager.

**CA-ACF2:** #SECRTT SECBY= is equivalent to @GRCE VALIDTE=. #SECRTT SECBY=EXT is equivalent to VALIDTE=YES, and #SECRTT SECBY=INT or OFF is equivalent to VALIDTE=NO.

**Subschema and area security:** Under CA-IDMS 12.0, there are two possible approaches to securing subschemas and areas using existing rules in the external security manager:

- Activate database security and supply an SRTT entry for run units (schemas) and areas.  
This approach allows safe-listing only at the database level but does not require a user exit.
- Define your own resource types for subschema and area in the SRTT and associate them with the corresponding predefined resources in the external security manager.  
This approach allows safe-listing using occurrence overrides but requires user exit 14.

Detailed information about subschema and area security is presented in the sections later in this appendix that are specific to each external security manager.

## D.3 CA-ACF2 conversion

**About this section:** This section contains information for administrators at sites where CA-ACF2 is used to protect CA-IDMS resources on a 10.2 system that is to be upgraded to 12.0.

### D.3.1 Administration

**Specifying security options:** For CA-IDMS 12.0, MUSOPT (in the address logonid record) is no longer used to specify CA-IDMS security options to CA-ACF2. Security options are specified in the SRTT (RHDCSRTT), which is loaded at start-up as part of the nucleus. Therefore, RHDCSRTT:

- Should be in the CDMSLIB concatenation
- May be LPA-resident

**Resource classes:** @GRCE INTTYPE corresponds to #SECRTT parameter RESTYPE=. This table presents the closest CA-IDMS correspondences to pre-defined resources in CA-ACF2:

CA-ACF2 @GRCE INTTYPE	CA-IDMS #SECRTT RESTYPE=
PGM, PGN	SPGM
DAT	AREA
TSK	TASK
SSC	NRU

@GRCE INSTYPE= corresponds to #SECRTT EXTCLS=. Additional mapping in CA-ACF2 may be required to relate EXTCLS to a specific rule type.

►► For more information about mapping external classes to ACF2 rule type codes, refer to *CA-ACF2 Administrator Guide*.

**Resource Rule Directories:** The RULES parameter of the GRCE macro is not supported in CA-IDMS 12.0. If you want resource rule directories to be globally resident, you must explicitly make a change to the CA-ACF2 OS/390 Global System options to designate them as globally resident. If you designate resource rule directories to be locally resident, there is currently no way in CA-IDMS 12.0 to reload or refresh them.

**External resource names:** For CA-ACF2 5.2, be sure resource names will not exceed 40 characters. Define rules to match resource names specified in the SRTT.

For example, assume the #SECRTT entry for resource type TASK was coded:

```
#SECRTT RETYPE=TASK,  
        EXTCLS='TSK',  
        EXTNAME=(SYSTEM,RESNAME),  
        SECBY=EXTERNAL
```

The format for TASK resource names presented to CA-ACF2 for validation will be *system-identifier.task-code*. If the system identifier is PROD, and PAYU is the task code to be protected, code the rule in CA-ACF2/MVS 5.2:

```
ACF  
SET RESOURCE(TSK)  
COMPILE *  
$KEY(PROD.PAYU) TYPE(TSK)  
UID(-) ALLOW
```

For CA-ACF2 6.0, the rule could be coded:

```
ACF  
SET RESOURCE(TSK)  
COMPILE *  
$KEY(PROD) TYPE(TSK)  
PAYU UID(-) ALLOW
```

In the above example, the key is combined with the resource name on the rule line to produce a resource name of PROD.PAYU This is one approach to creating a definition that allows resource names longer than 40 characters.

**Using existing rules:** To use existing CA-ACF2 rules, code #SECRTT entries in this form:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=resource-type,  
        EXTCLS='external-class',  
        EXTNAME=RESNAME,  
        SECBY=EXTERNAL
```

For example, assume you have defined this rule:

```
ACF  
T RESOURCE(TSK)  
COMPILE *  
$KEY(PAYU) TYPE(TSK)  
UID(-) ALLOW
```

The SRTT entry to use this rule would be:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=TASK,  
        EXTCLS='TSK',  
        EXTNAME=RESNAME,  
        SECBY=EXTERNAL
```

**Environment integrity:** Since security is now an integral part of the CA-IDMS/DC nucleus, no RFTIME definition is needed in CA-ACF2.

CA-IDMS control blocks can be protected via CA-IDMS/DC storage protection.

**Defining security schemes by region:** If you have multiple central versions and need to implement a different security scheme for each one, you can define a separate #SECRTT macro for each central version. This is a logical equivalent to the functionality provided by the IDMS MODE setting parameter. You can also choose to do any of the following in your CA-ACF2 environment to implement a different security scheme for multiple central versions:

- Write general allow-all rule
- Write extensive safe lists
- Use special CA-ACF2 OS/390 exit to turn off validation

**How to define minilid:** To define minilid, be sure to:

- Use ACFFDR instead
- Include @MUSASS for each IDMS region logonid
- Insure that each @MUSASS definition refers to an appropriate minilid definition

►► For complete information on defining minilid, refer to the *CA-ACF2 OS/390 Systems Programmer Guide*.

## D.3.2 Subschema security

**Using database security:** To secure subschemas using database security, you create SRTT entries similar to these examples:

1. Secure databases externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=DB,  
        SECBY=EXTERNAL,  
        .  
        .  
        .
```

2. Define run unit information for use by external security:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=NRU,  
        SECBY=EXTERNAL,  
        EXTCLS='SSC',  
        EXTNAME=(SSNAME)
```

By securing databases externally, you cause a security check to be routed to CA-ACF2 before a run unit is started. By specifying EXTNAME=(SSNAME) on the SRTT entry for NRU, you direct CA-IDMS to send the subschema name to CA-ACF2 to check the user's authorization to access the resource.

**Using user exit 14:** To secure subschemas with a user-defined resource, you create SRTT entries similar to these examples:

1. Define the resource and secure it externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=SSC,  
        SECBY=EXTERNAL,  
        EXTCLS='SSC',  
        EXTNAME=(RESNAME)
```

2. Create a safe list by defining one or more occurrence overrides:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=SSC,  
        SECBY=OFF,  
        RESNAME=subschema-name
```

Because the SSC resource type is user-defined, you are able to deactivate security for specific subschemas by specifying SECBY=OFF on the occurrence overrides. Security checking is performed by user exit 14, which is called on a BIND RUN UNIT.

►► For more information about user exit 14:

- See D.5, “Sample user exit 14 processing” on page D-19
- *CA-IDMS System Operations*

### D.3.3 Area security

**What CA-IDMS provides:** CA-IDMS protects areas when DB security is activated. This includes protection against access through CA-IDMS utilities, such as FORMAT, FIX PAGE, and UNLOCK. When areas are secured externally, READ or UPDATE authority is required for access. Create an SRTT entry for the AREA resource type to secure specify the external resource class for areas:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=AREA,  
        SECBY=EXTERNAL,  
        EXTCLS='DAT',  
        EXTNAME=(RESNAME)
```

All areas in all databases are secured except for areas in a database for which an occurrence override deactivates security. In this case, no areas of the database are secured. Thus, safe-listing is possible only at the database level.

**Securing individual areas:** You can use existing CA-ACF2 rules and secure individual areas by defining a resource type, securing it, associating it with the pre-defined 'DAT' resource in CA-ACF2, and then specifying occurrence overrides, as in these examples:

1. Define the resource and secure it externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=DAT,  
        SECBY=EXTERNAL,  
        EXTCLS='DAT',  
        EXTNAME=(RESNAME)
```

**Note:** RESTYPE=DAT names DAT as a user-defined resource to CA-IDMS; however, any 4-character name not already defined in the SRTT would be valid. The association with the pre-defined resource 'DAT' in CA-ACF2 is made by EXTCLS='DAT'.

2. Create a safe-list by specifying one or more occurrence overrides:

```
#SECRTT TYPE=OCCURRENCE,  
        RESTYPE=DAT,  
        SECBY=OFF,  
        RESNAME=area-name
```

By defining the DAT resource type to CA-IDMS and securing it externally, you are able to deactivate security for specific area occurrences. Security checking is performed by user exit 14, which is called on a READY AREA.

►► For more information about user exit 14:

- See D.5, "Sample user exit 14 processing" on page D-19
- *CA-IDMS System Operations*

### D.3.4 CA-ACF2 CA-IDMS interface

The status of the security functions supported by the CA-ACF2 CA-IDMS 10.2 interface in the CA-IDMS 12.0 environment is summarized below:

CA-ACF2 CA-IDMS function	Status in CA-IDMS 12.0
ACMSRSON	Supported by the #SECSGON macro
ACMSRSOF	Supported by the #SECSGOF macro
ACMSRRSV	Supported by the #SECHECK macro
ACMSRRLD	No comparable support
ACMSRRDD	No comparable support
ACMSRDLU	No comparable support
ACMSRPRV	Beginning with the 9212 maintenance release for CA-IDMS 12.0, supported through centralized security, but not by a macro. However, client can explicitly code a call to CA-ACF2 using ACFSVC with ACVALD or by using HLI.
ACMSRCVT	No comparable support
ACMSRCTL	No comparable support

**Note:** The functions listed above are described in the CA-ACF2 DSECT, ACMSERV.

- For more information about #SECSGON, #SECSGOF, and #SECHECK see *CA-IDMS Security Administration*.
- For more information on using ACFSVC or HLI, refer to the *CA-ACF2 Systems Programmer Guide* for the appropriate operating system.
- For more information on the CA-ACF2 CA-IDMS interface, refer to the *CA-ACF2 CA-IDMS Support Guide* for the appropriate operating system.

## D.4 CA-Top Secret conversion

**About this section:** This section contains information for administrators at sites where CA-Top Secret is used to protect CA-IDMS resources on a 10.2 system that is to be upgraded to 12.0.

### D.4.1 Administration

**Resource classes:** This table presents #SECRTT parameter RESTYPE= values that most closely correspond to pre-defined CA-Top Secret resource classes:

CA-Top Secret resource class	CA-IDMS #SECRTT RESTYPE=
PROGRAM	SPGM
AREA	AREA
LCF	TASK
SUBSCHEM	NRU

**External resource names:** CA-Top Secret resource names cannot exceed 44 characters; therefore, in constructing external resource names using the #SECRTT EXTNAME parameter, be sure an occurrence of the resource name could not exceed this limit. Keep in mind that:

- In CA-IDMS, SQL resource names may be as many as 18 characters
- Ownership of CA-Top Secret resources is defined using 1- to 8-character prefixes

**Using existing rules:** To use existing CA-Top Secret rules, code #SECRTT entries in this form:

```
#SECRTT TYPE=ENTRY,
  RESTYPE=resource-type,
  EXTCLS='external-class' ,
  EXTNAME=RESNAME,
  SECBY=EXT
```

For example, assume you have defined this rule:

```
TSS ADD(USER01) PROGRAM (TSTAA455)
TSS PERMIT(USER22) PROGRAM (TSTAA455)
```

The SRTT entry to use this rule would be:

```
#SECRTT TYPE=ENTRY,
  RESTYPE=SPGM,
  EXTCLS='PROGRAM',
  EXTNAME=RESNAME,
  SECBY=EXTERNAL
```

## D.4.2 Task security

**Using external security:** To secure tasks using external security, you create SRTT entries similar to these examples:

1. Secure tasks externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=TASK,  
        SECBY=EXTERNAL,  
        EXTCLS='LCF',  
        EXTNAME=(RESNAME)
```

2. Create occurrence entries for any "safe" tasks. For example, public tasks or custom signon tasks:

```
#SECRTT TYPE=OCCURRENCE,  
        RESTYPE=TASK,  
        SECBY=OFF,  
        RESNAME=ANY1
```

Coding an EXTCLS of LCF provides both an OTRAN and LCF type check when running TSS4.3. In contrast, when running TSS4.2 only an LCF type check is provided. If you want an OTRAN type check performed, then code EXTCLS=OTRAN.

**CA-Top Secret definitions:** OTRAN protected tasks are owned and permitted like normal resources. For example:

```
TSS ADD(acid) OTRAN(PAY1)  
TSS PERMIT(acid) OTRAN (PAY1)
```

LCF protection is provided by defining either an inclusive list (TRANS) or an exclusive list (XTRANS), but not both.

The example below shows the definition of an inclusive list, allowing a user access to only the transactions listed.

```
TSS ADD(acid) TRANS(IDMSPROD,(PAY1,PAY2))
```

The example below shows the definition of an exclusive list which denies a user access to the transactions listed.

```
TSS ADD(acid) XTRANS(IDMSPROD,(PAY5,PAY6))
```

## D.4.3 Subschema security

**Using database security:** To secure subschemas using database security, you create SRTT entries similar to these examples:

1. Secure databases externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=DB,  
        SECBY=EXTERNAL,  
        .  
        .  
        .
```

2. Define run unit information for use by external security:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=NRU,  
        SECBY=EXTERNAL,  
        EXTCLS='SUBSCHEM',  
        EXTNAME=(SSNAME)
```

By securing databases externally, you cause a security check to be routed to CA-Top Secret before a run unit is started. By specifying EXTNAME=(SSNAME) on the SRTT entry for NRU, you direct CA-IDMS to send the subschema name to CA-Top Secret to check the user's authorization to access the resource.

**Using user exit 14:** To secure subschemas with a user-defined resource, you create SRTT entries similar to these examples:

1. Define the resource and secure it externally:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=SSCH,  
        SECBY=EXTERNAL,  
        EXTCLS='SUBSCHEM',  
        EXTNAME=(RESNAME)
```

This entry defines SSCH as a resource type in CA-IDMS and associates it (through the EXTCLS= parameter) with the pre-defined SUBSCHEM resource in CA-Top Secret.

2. Create a safe list by specifying one or more occurrence overrides:

```
#SECRTT TYPE=ENTRY,  
        RESTYPE=SSCH,  
        SECBY=OFF,  
        RESNAME=subschema-name
```

Since the SSCH resource type is user-defined, you are able to specify occurrence overrides for it. Overrides that specify SECBY=OFF effectively deactivate security for the specified subschema occurrences. Security checking is performed by user exit 14, which is called on a BIND RUN UNIT.

►► For more information about user exit 14:

- See D.5, "Sample user exit 14 processing" on page D-19
- *CA-IDMS System Operations*

**CA-Top Secret definitions:** In CA-Top Secret you establish ownership of a subschema and grant access to the subschema, as in these examples:

```
TSS ADD(acid) SUBSCHEM(subschema-name)
TSS PERMIT(acid) SUBSCHEM(subschema-name)
```

#### D.4.4 Area security

**What CA-IDMS provides:** CA-IDMS protects areas when DB security is activated. This includes protection against access through CA-IDMS utilities, such as FORMAT, FIX PAGE, and UNLOCK. When areas are secured externally, READ or UPDATE authority is required for access. Create an SRTT entry for the AREA resource type to secure specify the external resource class for areas:

```
#SECRTT TYPE=ENTRY,
        RESTYPE=AREA,
        SECBY=EXTERNAL,
        EXTCLS='AREA',
        EXTNAME=(RESNAME)
```

All areas in all databases are secured except for areas in a database for which an occurrence override deactivates security. In this case, no areas of the database are secured. Thus, safe-listing is possible only at the database level.

**Securing individual areas:** You can use existing CA-Top Secret rules and secure individual areas by defining a resource type, securing it, associating it with the pre-defined 'AREA' resource in CA-Top Secret, and then specifying occurrence overrides, as in these examples:

1. Define the resource and secure it externally:

```
#SECRTT TYPE=ENTRY,
        RESTYPE=IDAR,
        SECBY=EXTERNAL,
        EXTCLS='AREA',
        EXTNAME=(RESNAME)
```

2. Create a safe list by specifying one or more occurrence overrides:

```
#SECRTT TYPE=ENTRY,
        RESTYPE=IDAR,
        SECBY=OFF,
        RESNAME=area-name
```

Since the IDAR resource type is user-defined, you are able to specify occurrence overrides for it. Overrides that specify SECBY=OFF effectively deactivate security for the specified area occurrences. Security checking is performed by user exit 14, which is called on a READY AREA.

►► For more information about user exit 14:

- See D.5, “Sample user exit 14 processing” on page D-19
- *CA-IDMS System Operations*

## D.5 Sample user exit 14 processing

The examples below are intended to show you how to use exit 14 and are not intended to be used as actual executable routines.

**Subschema example:** This code presents an example of validating access to a subschema resource (an occurrence of user-defined resource type SSC) on a BIND RUN UNIT using user exit 14:

```

SECWORK DSECT
WKAUTHB DS CL6
WKSRB DS CL(SRBSCLEN)
SECWKLFN EQU ((*-SECWORK+3)/4)*4
:
LM R6,R7,0(R1)
USING SSC,R7
C R6,=F'59'
BE E14BND
C R6,=F'97'
BNE E14RDY
XR R15,R15
E14RTN DS 0H
#RTN
*-----
* Request is BIND RUN UNIT
*-----
E14BND DS 0H
#GETSTK =SECWKLFN,REG=(R6)
USING SECWORK,R6
*-----
* Set status to 1469 if security violation. Site may choose to
* return a status other than 1469.
*-----
L R2,SSCIDBCM+16
#SECHECK SRB=WKSRB,
RESNAME='SSC',
RESNAME=(R2),
AUTHRTY=EXECUTE,
RGSV=(R2-R8)
X
X
X
X
LTR R15,R15
BZ E14RTN
MVC SSCSTAT,=C'1469'
B E14RT

```

**Area example:** This code presents an example of validating access to an area resource (an occurrence of user-defined resource type DAT) on a READY AREA using user exit 14:

## D.5 Sample user exit 14 processing

---

```
*-----  
* Request is READY AREA of some sort  
*-----  
E14RDY DS 0H  
#GETSTK =SECWKLNF,REG=(R6)  
USING SECWORK,R6  
MVC WKAUTHB,CSANULLS  
*-----  
* If retrieval usage mode  
*-----  
OI WKAUTHB,SAARSELM  
B E14SCHKA  
*-----  
* If update usage mode  
*-----  
OI WKAUTB,SAARUPDM  
*-----  
* Set status 0966 if security violation. Site may choose to return  
* a status other than 0966.  
*-----  
E14SCHKA DS 0H  
L R2,SSCIDBCM+12 X  
#SECHECK SRB=WKSRB, X  
RESTYPE='DAT', X  
RESNAME=(R2), X  
AUTHRTY=WKAUTHB, X  
RGSV=(R2-R8)  
LTR R15,R15  
BZ E14RTN  
MVC SSCSTAT,=C'0966'  
B E14RTN
```

## **Appendix E. 10.2 DSECT Replacement**

---

E.1 About this appendix . . . . .	E-3
E.2 New 12.0 macros . . . . .	E-4



## **E.1 About this appendix**

With CA-IDMS 12.0, you must generate several database-related DSECTs using macro statements rather than COPY books. This appendix lists 10.2 COPY books replaced by new macros in 12.0.

## E.2 New 12.0 macros

**Macros replacing 10.2 COPY books:** The table below identifies the 12.0 macros which replace 10.2 COPY books. New 12.0 macros which you must generate by a macro statement are also listed.

If you have programs that reference the 10.2 COPY book, replace the COPY book with the appropriate 12.0 macro.

10.2 COPY book	New macro	Comments
COPY #DMCEQ		Obsolete
COPY #DMCDS	#DMCLDS	All DMCL blocks
None	#SUBCB VARS=YES	All SUBSCHEMA blocks
COPY #SUBEQ	#SUBCB VARS=YES, TYPE=EQUATES	All SUBSCHEMA blocks using EQUs
COPY #AFMDS	#DMCLDS TYPE=AFMDS	FM61 block
COPY #BCRDS	#DMCLDS TYPE=BCRDS	BC53 block
COPY #DPRDS	#DMCLDS TYPE=DPRDS	PR60 block
COPY #FCBDS	#DMCLDS TYPE=FCBDS	FC59 block
COPY #JBCDS	#DMCLDS TYPE=JBCDS	JB63 block
COPY #JCBDS	#DMCLDS TYPE=JCBDS	JD62 block
None	#DMCLDS TYPE=SEGDS	SG49 block
None	#DMCLDS TYPE=SYMDS	SY40 block
COPY #VACDS	#FACDS VARS=YES	AC56 and VAC blocks
COPY #VCRDS	#FCRDS VARS=YES	CR55 and VCR blocks
None	#FFKDS VARS=YES	FK76 and VFK blocks
COPY #VMRDS	#FMRDS VARS=YES	MR53 and VMR blocks
COPY #VORDS	#FORDS VARS=YES	OR52 and VOR blocks
COPY #VPCDS	#FPCDS VARS=YES	PC70 and VPC blocks
None	#FSADS VARS=YES	SA73 and VSA blocks
COPY #VSRDS	#FSRDS VARS=YES	SR51 and VSR blocks
COPY #VLRDS	#FLRDS VARS=YES	LR80 and VLR blocks
COPY #VLQDS	#FLQDS VARS=YES	LQ81 and VLQ blocks
COPY #VPSDS	#FPSDS VARS=YES	PS83 and VPS blocks
COPY #FACDS	#FACDS	AC56 block

<b>10.2 COPY book</b>	<b>New macro</b>	<b>Comments</b>
COPY #FAMDS	#FAMDS	AM57 block
COPY #FAPDS	#FAPDS	AP72 block
COPY #FCRDS	#FCRDS	CR55 block
COPY #FFDDS	#FFDDS	FD54 block
COPY #FIBDS	#FIBDS	IB50 block
None	#FFKDS	FK76 block
None	#FLBDS	LB74 block
COPY #FMRDS	#FMRDS	MR53 block
COPY #FORDS	#FORDS	OR52 block
COPY #FPCDS	#FPCDS	PC70 block
COPY #FRPDS	#FRPDS	RP71 block
None	#FSADS	SA73 block
COPY #FSRDS	#FSRDS	SR51 block
COPY #FLRDS	#FLRDS	LR80 block
COPY #FLQDS	#FLQDS	LQ81 block
COPY #FPHDS	#FPHDS	PH85 block
COPY #FPSDS	#FPSDS	PS83 block
COPY #FSDDS	#FSDDS	SD82 block

## E.2 New 12.0 macros

---

# Index

---

## Special Characters

#DCPARM macro 5-4

## A

application conversion  
activities required 9-3  
CA-ADS 9-13  
deleted CA-IDMS modules 9-5  
relinking programs 9-8  
SPF indexes 9-11  
SYSIDMS parameter file 9-4  
ASF extent areas  
IDMSRSSM program 8-5  
IDMSRUPD program 8-14  
migration of 8-3—8-22  
updating IDMSR schema 8-20

## C

CA-ACF2  
*See also* external security manager  
@GRCE INSTYPE D-9  
@GRCE INTTYPE D-9  
area security D-12  
AUTHFLD in the logonid record D-6  
CA-IDMS interface D-13  
IDMS MODE setting D-11  
minilid D-11  
resource names D-9  
RFTIME definition D-10  
safe lists D-12, D-13  
specifying CA-IDMS security options to D-9  
subschema security D-11  
CA-ADS upward compatibility 9-13  
CA-Top Secret  
*See also* external security manager  
area security D-18  
multiple signons D-5  
resource classes D-15  
resource names D-15  
safe lists D-17, D-18  
subschema security D-16  
task security D-16  
VTHRESH option D-7  
CAICCI 5-11

central security  
area security D-8, D-12, D-18  
as part of CA-IDMS/DC nucleus D-10  
database security D-11, D-12, D-16, D-18  
default logon ID D-4  
external resource names D-9, D-15  
external security class D-9  
multiple signons D-5  
occurrence overrides D-7, D-12, D-13, D-17, D-18  
resource access violation threshold D-7  
resource types D-9, D-15  
RHDCSNON D-4  
securing multiple central versions D-11  
securing tasks using external security D-16  
security class codes D-6  
signon security D-6  
specifying external security enforcement D-7  
SRTT D-7  
subschema security D-8, D-11, D-16  
user exit 14 D-11, D-13, D-17, D-18, D-19  
user exits D-4  
COBOL precompiler  
JCL changes 9-7  
conversion activities  
before you begin 1-12  
table of 1-4

## D

database name table 2-25, 2-29  
default dictionary 2-23  
defining 2-25—2-29  
defining DBNAMES 2-25  
differences between 10.2 and 12.0 2-25  
grouping segments 2-24  
mixed page groups 2-29  
R120DBTB listing B-6  
subschema mapping 2-25  
database procedures  
changes 7-5  
DDLCDLOG area  
formatting 5-13  
DDLCDRUN area  
formatting 5-13  
DDLCDSCR area  
formatting 5-13  
default dictionary 2-23

---

dictionary 4-5, 4-16  
area changes 4-4  
DDLCAT area 4-4  
DDLCATLOD area 4-4  
DDLCATX area 4-4  
files A-3  
migration of 4-5—4-16  
RHDCMIG1 utility 4-4  
RHDCMIG2 utility 4-4  
segments A-3  
set up of application dictionary 4-20  
set up of system dictionary 4-19

DMCL  
R120DMCL listing B-4  
DMCL conversion  
database name table 2-23  
DMLC syntax generator 2-6  
process 2-4  
DMCL syntax generator  
execution JCL 2-9  
output 2-12  
purpose 2-7  
syntax 2-8

## E

extent areas  
*See also* ASF extent areas  
changes 8-3  
DDLDCQUE area 8-11  
external request units 5-11  
external security manager  
area security D-8  
automatic terminal signon D-4  
how it is specified to CA-IDMS D-7  
password control D-4  
password reverification D-4  
resource access violation threshold D-7  
safe lists D-7  
subschema security D-8  
validation of access to CA-IDMS resources D-7  
external user sessions 5-11

## I

IDMS MODE setting D-11  
IDMSDIRL utility 4-17  
IDMSDMLC  
*See* COBOL precompiler  
IDMSDMLP  
*See* PL/I precompiler

IDMSQSAM 9-5  
IDMSRSSD subschema 8-13  
IDMSRSSM program  
*See also* ASF extent areas  
execution JCL 8-6  
output 8-9—8-13  
purpose 8-5  
syntax 8-5  
IDMSRUPD program 8-14—8-19  
*See also* ASF extent areas  
execution JCL 8-15  
output 8-18  
purpose 8-14  
syntax 8-14  
IDMSSCON utility 4-22  
*See also* subschema regeneration  
execution JCL 4-22  
input 4-22  
IDMSTRAC 9-5  
IDMSUTIO 9-6  
installed 12.0 test environment  
DBTABLE listing B-5  
description 1-11  
DMCL listing B-3  
segments and files A-3

## J

JCL  
DMCL syntax generator 2-9  
IDMSCHEM program 8-20  
IDMSRSSM program 8-6  
IDMSRUPD program 8-15  
IDMSSCON utility 4-22  
RHDCMIG1 dictionary migration utility 4-8  
RHDCMIG2 dictionary migration utility 4-13  
RHDCSMIG security migration utility 3-7  
JCL changes  
CA-ADS/Batch 9-4  
CA-CULPRIT 9-4  
CA-OLQ 9-4  
COBOL precompiler 9-7  
PL/I precompiler 9-7  
region size 9-12  
security 9-4  
SYSIDMS parameter file 9-4  
journal files  
formatting 5-13

---

## P

PL/I precompiler  
JCL changes 9-7

## R

R120DBTB listing  
R120DMCL listing  
region size  
relinking application programs 9-8  
Batch programs in a VSE/ESA environment 9-8  
DC-BATCH programs 9-8  
IDMS-DC COBOL programs 9-8  
IDMS-DC PL/I programs 9-8  
RHDCMIG1 dictionary migration utility  
execution JCL 4-8  
purpose 4-5  
RHDCMIG2 dictionary migration utility  
execution JCL 4-12  
purpose 4-5  
RHDCSMIG security migration utility  
execution JCL 3-7  
input 3-7  
output 3-13—3-19

## S

schema changes  
security definition  
migration of 3-4—3-22  
RHDCSMIG security migration utility 3-7  
security, area D-19  
security, subschema D-19  
specifying DBNAMES 2-25  
SPF indexes 9-11  
subschema mapping 2-25  
subschema regeneration 4-21—4-27  
IDMSSCON utility 4-22  
options 4-21  
SVC 5-4  
SYSIDMS parameter file 9-4  
system generation  
about regenerating definitions 5-4  
changes to TASK and PROGRAM statements 5-9  
migrated system definition 5-5  
SVC 5-4  
system definition changes 5-5—5-9  
system startup  
#DCPARM macro 5-4  
changes 5-14  
preparing to startup 5-13

## T

TAT table 4-18

## U

user exit 14 D-19  
user exits  
changes 7-4  
new exits 7-4  
utility statements  
changes to utility programs 6-6  
execution JCL 6-5  
new statements 6-4  
PUNCH statement 6-5  
security 6-8  
SYSIDMS parameter file 6-7

